MS OFFICE ACCESS 95-2016

V28.0 Revision 1002, {oUUID 1.680}

 Please consider the environment - do you really need to print this document!?

Vincent Isoz

2018-10-02

Remarques:

Pour qu'il soit utilisable d'une manière rationnelle et sans danger, ce support qui constitue un "super condensé" d'un exposé qui tiendrait très facilement sur plusieurs milliers de pages (voir les ouvrages de cette taille disponible sur le commerce) doit absolument être complété par de nombreuses notes et exposés oraux, au cours desquels les notions nouvelles sont présentées au moyen de situations concrètes et illustrées par de nombreux exemples dont le choix dépend essentiellement du déroulement de la formation afin d'exciter l'esprit critique des apprenants. Ce support correspond à une formation d'environ 15 jours à 6.5 heures par jour pour un groupe de 6 personnes.

Ce qui est vu dans ce support peut être appliqué à toutes les versions antérieures à MS Access 2007. Je ne mettrai donc pas ce document à jour pour qu'il corresponde à la version 2007, 2010 ou 2012 (sauf pour les nouveautés) puisque Microsoft à la tendance ces dernières années à changer l'interface à chaque version et j'ai autre chose à faire de mon temps que de mettre à jour des captures d'écran.

Je tiens également à m'excuser du fait que ce document mélange les captures d'écran d'un grand nombre de versions de MS Access (97 à 2010) et en plus de différentes langues (français + anglais). Effectivement, mon métier m'oblige constamment à changer d'ordinateur et ainsi le contenu des documents que je rédige. J'espère que le lecteur comprendra en attendant une uniformisation.

Il y a de nombreuses marques déposées qui sont nommées dans le présent support. Plutôt que d'utiliser le symbole du trademark sur chaque occurrence de marque nommée, j'ai choisi d'utiliser le nom seul uniquement dans un souci d'esthétique éditoriale (ce qui devrait aussi bénéficier au propriétaire de la marque), sans aucune intention de violer une quelconque réglementation ou législation.

Pour terminer, je voudrais remercier ici les quelques collègues (Olivier Weber, Fabrice Fournier) et clients qui ont bien voulu me faire part de leurs remarques pour améliorer le contenu de ce livre électronique. Il est cependant certain qu'il est encore perfectible sur de nombreux points.

Si vous souhaitez être informé des nouvelles versions **majeures** de ce document n'hésitez pas à m'écrire un mail dans ce sens: isoz@sciences.ch.

TABLE DES MATIÈRES

Contents

TABLE DES MATIÈRES ... *2*
1 Liens internet ... *10*
1. À propos du rédacteur ... *11*
2 Introduction ... *12*
2.1 Environnement ... 13
2.2 Protocole ... 15
2.3 MS Excel (tableur) VS MS Access (SGBDR) ... 16
2.4 Évolution de l'interface ... 17
3 Notions de base de données ... *20*
3.1 Types de BDD ... 20
3.2 Vues d'une base de données ... 21
3.3 Modules d'un SGBDR ... 24
3.4 Méthodes de création de BDD ... 25
3.5 Modèle entité-relation (MER) ... 27
3.6 Normalisation ... 30
3.6.1 Première forme normale (1FN) ... 32
3.6.2 Deuxième forme normale (2FN) ... 33
3.6.3 Troisième forme normale (3FN) ... 34
3.6.4 Quatrième forme normale ... 35
3.6.5 Exercice ... 36
3.7 Modélisation de BDD avec MS VISIO ... 36
3.8 Nomenclature de Leszynski/Reddick ... 44
4 Tables ... *47*
4.1 Liaison MS Excel/CSV ... 49
4.2 Création de tables ... 51
4.3 Formulaire simple (auto-form) ... 52
4.4 Contrôle des données ... 53
4.4.1 Légende ... 54
4.4.2 Types de données (Typages) ... 55
4.4.3 Formats ... 64
4.4.4 Masques de saisie ... 65
4.4.5 Validation (Valide Si) ... 68
4.5 Import MS Excel ... 72
4.6 Import MS Access ... 73
4.7 Import/liaison MS Outlook ... 74
4.8 Import HTML ... 77
4.9 Format des tables ... 81
4.10 Outil recherche ... 82
4.11 Propriétés des tables ... 83
4.12 Tris et filtres ... 84
4.12.1 Critères numériques et textes ... 85
4.12.2 Critères numériques et dates ... 85
4.12.3 Caractères génériques (wildcards) ... 85
4.12.4 Fonctions génériques ... 86
4.12.5 Tris et filtres avancés ... 87
5 Relations (jointures) ... *88*
5.1 Assistant liste de choix ... 88
5.1.1 Assistant liste de choix (ALC) statique ... 88

5.1.1.1 ALC statique à choix unique de type zone de liste déroulante 88
5.1.1.2 ALC statique à choix unique de type zone de liste..91
5.1.1.3 ALC statique à choix unique de type liste déroulante extensible 93
5.1.1.4 ALC statique à choix multiple de type liste déroulante extensible...............................95
5.1.2 Assistant liste de choix (ALC) lié.. 97
5.1.2.1 ALC statique lié à choix unique de type zone de liste déroulante 97
5.2 Relation un à un 102
5.3 Relation un à plusieurs... 103
5.4 Relation plusieurs à plusieurs 107
5.5 Relation "simple" (sans intégrité référentielle)... 107
5.6 Relation avec intégrité référentielle.. 107
5.7 Relation circulaire (auto-liaison) 113
5.8 Relations d'héritage et composites.. 115
5.9 Index simples et combinés.. 116
6 Formulaires (simples) 119
6.1 Choix de la méthode d'affichage des formulaires... 119
6.2 Filtre par formulaire.. 121
6.3 Formulaire en mode design... 125
6.4 Création de boutons de formulaires 125
6.5 Filtres requêtes... 132
6.6 Outil recherche.. 132
6.7 ComboBox de recherche... 133
6.8 Groupes d'options 135
6.9 Champs calculés 137
6.10 ListBox de recherche 137
6.11 Fonction DSum (SomDom en FR) 138
6.12 Fonction DCount (CpteDom en FR).. 139
6.13 Fonction IIf (formulaire)... 140
6.14 Fonctions d'environnement.. 140
6.15 Fonction DLookUp (RechDom en FR) 140
6.15.1 Équivalent de DLookUp en cas d'utilisation de GUID.................................... 141
6.16 Onglets... 143
6.17 Valeurs par défaut et filtres.. 145
7 Requêtes (simples)... 147
7.1 Optimisation des requêtes... 150
7.2 Requête simple (de projection).. 150
7.3 Tris dans les requêtes.. 151
7.4 Requête multitable sans liaisons 153
7.5 Requête multitable avec liaisons.. 154
7.6 Requête de distinction... 155
7.7 Cinq premiers.. 155
7.8 Requête avec critère.. 156
7.9 Requête concaténation dans liste de choix... 158
7.10 Colonne calculée... 160
7.11 Critères multiples... 160
7.12 Calcul de synthèse (d'agrégation) 162
7.13 Regroupement et calculs.. 162
7.14 Requête et macro d'export 164
7.15 Requête mise-à-jour... 167
7.16 Requête mise-à-jour (Rechercher/Remplacer).. 168
7.17 Requêtes de synthèse 170
7.18 Requête d'union 171
7.19 Requête d'intersection... 175
7.20 Requête de comptage 176
7.21 Requête d'union (bis) 177

7.22 Requêtes de requêtes ... 178
7.23 Divers (exercices) ... 180
8 États (Rapport)-Formulaires (complexes) ... 181
8.1 Carnet d'adresse ... 184
8.2 Synthèse et requête ... 186
8.3 Sous-états ... 188
8.4 Rapports avec groupes (pour lettres ou factures) ... 192
8.5 Rapport paramétré par formulaire ... 207
8.6 Objet ActiveX ... 208
8.7 Graphique statistique inséré ... 209
8.8 Graphique inséré ... 213
8.9 Tableaux croisés dynamiques ... 214
8.10 Graphiques croisés dynamiques ... 215
9 Requêtes (complexes avec ou sans VBA) ... 216
9.1 Requêtes avec jointures ... 216
9.2 Requête regroupement par premiers éléments ... 218
9.3 Requête avec critères ... 220
9.4 Requête d'analyse de fréquence (contingence) ... 221
9.5 Requête paramétrée ... 222
9.6 Requête d'ajout ... 224
9.7 Requête de suppression ... 225
9.8 Requête d'analyse croisée (avec assistant) ... 225
9.9 Requête d'analyse croisée (sans assistant) ... 226
9.10 Requête d'analyse croisée temporelle ... 227
9.11 Requête d'analyse croisée paramétrée ... 228
9.12 Requête (trouver les) doublons ... 229
9.13 Requête de suppression des doublons d'enregistrements ... 230
9.13.1 Première approche: non-destructive sur la table originale ... 230
9.13.2 Deuxième approche: Destructive sur la table d'origine ... 232
9.14 Requête de non correspondance ... 233
9.15 Requêtes de création/définition (mode SQL) ... 234
9.16 Requête de distribution en % sur comptage ... 236
9.17 Requête de (Pareto) distribution en % sur somme ... 237
9.18 Requête de cumul chronologique ... 238
9.19 Requête système ... 239
9.20 Requêtes de statistiques non natives ... 239
9.20.1 Requête de valeur modale ... 240
9.20.2 Requête d'obtention de la médiane ... 243
9.20.3 Requêtes d'intervalle de fluctuation ... 246
9.20.4 Requêtes moyenne mobile et somme cumulée ... 247
9.20.5 Requêtes de data mining/machine learning ... 249
9.21 Analyse simple de portefeuilles ... 249
9.22 Transactions ... 252
10 Fonctions ... 254
10.1 Relation d'ordre comme validation d'un record (table) ... 254
10.2 Fonctions Date comme valeur par défaut (tables) ... 255
10.3 Fonctions Now comme valeur par défaut (tables) ... 256
10.4 Fonctions d'environnement (tables) ... 256
10.5 Fonction textes Left, Right, Instr (requête) ... 257
10.6 Fonctions textes UCase, LCase, &, TRIM, StrConv (requête) ... 258
10.7 Fonctions replace et MID (requête) ... 260
10.8 Fonctions de dates Year, Day, Month (requête) ... 262
10.9 Fonction de formatage de dates et DatePart (requête) ... 263
10.10 Fonction de date DateDiff (requête) ... 265
10.11 Calculs de jours avec DateAdd (Formulaire) ... 265

10.12 Fonction Logique IIf (formulaire) 267
10.13 Fonctions d'arrondi Round (requête) 268
10.14 Fonction conditionnelle Switch (requête) 268
10.15 Fonctions IsNull et NZ (requête) 270
10.16 Fonction DSum (formulaire) 272
10.17 Fonction DCount (formulaire) 274
10.18 Fonction DLookUp (formulaire) 275
10.19 Fonctions Count et DCount (requête) 276
10.20 Fonction Dsum (requête) 277
10.21 Fonction Dsum et Dates (requête) 278
11 Interfacage de l'application 280
11.1 Propriétés des formulaires 280
11.2 Propriétés des champs de formulaires 282
11.3 Sous-formulaires (Assistant) 285
11.4 Champs calculés 286
11.5 Sous-formulaires (Boîte à outils contrôle) 287
11.5.1 Total d'un sous-formulaire dans le formulaire principal 289
11.6 Formatage 290
11.7 Champs OLE 292
11.8 Esthétique rapports 297
11.9 Formulaire de démarrage (switchboard) 298
11.10 Options de démarrage 300
11.11 Barre d'outils personnalisée 305
11.11.1 MS Access 2003 et antérieur 305
11.11.2 MS Access 2007 et ultérieur 306
11.11.2.1 Masquer le ruban au démarrage de MS Access 2007 306
11.11.2.2 Masquer le ruban au démarrage de MS Access 2010 307
12 Finitions élémentaires 308
12.1 Compactage 308
12.2 Protection par mot de passe 308
12.3 MS Query 309
12.4 Publipostage 310
13 Macros 311
13.1 Macros simples 312
13.1.1 Exécution de requêtes en mode création 312
13.1.2 Contrôle de saisie simple 313
13.1.3 Import de données 314
13.1.4 Contrôles sur formulaires 315
13.2 Groupe de macros 317
13.3 Groupe conditionnel de macros 318
13.4 Macro AutoExec 320
13.5 Macros Run Code 321
13.6 Macros Import/Export 323
13.7 Macro ReQuery 323
13.8 Groupe de macros simple 325
13.9 Groupe de macros complexe 328
13.1 Conversion de Macros en VBA 330
14 Théorie internet / intranet 332
14.1 Formulaire web 332
14.2 Application Web avec SharePoint 2010/Access 2010 335
14.3 Application Web avec SharePoint 2013/Access 2013) 339
14.4 Application Web avec Office 365 345
15 Optimisation et analyse 346
15.1 Table 346

15.2 Performances 346
15.3 Documenter..... 348
16 Distribution 349
16.1 Fractionnement d'une base..... 350
16.2 Réseau..... 352
17 Securité avancée..... 354
17.1 Définition d'un mot de passe..... 354
17.2 Sécurité au niveau utilisateur..... 356
17.2.1 Protocole..... 356
17.2.2 Méthodes 357
17.2.3 Modélisation des droits..... 359
17.2.4 Dangers de l'espace de travail "par défaut": 360
17.2.5 Un peu de VBA avec la sécurité..... 377
17.2.6 A propos du fichier ldb..... 380
17.2.7 Problèmes de sécurité avec les requêtes 380
17.3 Déploiement..... 381
18 Synchronisation (réplication)..... 383
18.1 Types de réplicas 383
18.2 Création d'un réplica maître..... 383
18.3 Création d'un réplica partiel..... 389
18.4 Modifications sur les objets de la base de données..... 390
18.5 Comparatif Maitre-Réplica lors de la synchronisation 393
18.6 3 Conflits rencontrés lors de la synchronisation des bases 394
18.6.1 Subtilités de prédominance du maître..... 394
19 Visual Basic Application 396
19.1 Objectifs..... 398
19.2 Historique 398
19.3 Types de données..... 399
19.4 Nomenclature de Lezsynski-Reddick 402
19.5 Commentaires..... 403
19.6 Table des objets VBA et table ASCII..... 405
19.7 Prise en main du V.B.A 408
19.7.1 Exemples génériques à Office 410
19.8 Appliquer le filtre par formulaire automatiquement..... 414
19.9 Appliquer un filtre avec critères sur un formulaire..... 414
19.10 Détection formulaire ouvert..... 415
19.11 Détection de la version et compactage..... 415
19.12 Création d'un bouton ouvrant une page web..... 415
19.13 Exécution de code SQL (via VBA) 416
19.13.1 Création de table..... 416
19.13.2 Retour d'une requête 416
19.13.3 Boucler sur les enregistrements d'une requête..... 416
19.13.4 Source dynamique de formulaire..... 417
19.13.5 Création d'une table 417
19.13.5.1 Suppression de toutes les tables.....419
19.13.6 Ajout de nouveaux enregistrements..... 419
19.14 Imports/Exports 419
19.14.1 Import de données de MS Excel..... 419
19.14.2 Export paramétré d'un fichier MS Excel 420
19.14.3 Import d'un fichier CSV 420
19.15 Automatiser la mise à jour des tables de type BDD liées 423
19.16 Champ de Recherche sur formulaire..... 425
19.17 Evénement sur sortie de formulaire (validation des champs de saisie) 426
19.18 Evénement sur fermeture de formulaire (validation des champs de saisie)..... 427

19.19 Focus sur formulaire 428
19.20 Filtres 430
19.20.1 Filtre paramétré par bouton 430
19.20.2 Filtre paramétré par une zone d'options 432
19.20.3 Adaptation au filtre par formulaire 433
19.20.4 Filtres avec boutons a bascule 433
19.21 Listes déroulantes 434
19.21.1 Ouvrir une liste déroulante lors du survol avec la souris 434
19.21.2 Filtrage d'une liste déroulante sur la base d'un champ 436
19.21.3 Filtrage d'une liste déroulante sur la de la saisie dans la liste déroulante 436
19.21.4 Ajout directe d'une données dans une liste déroulante 438
19.21.5 Liste déroulante à choix multiples 442
19.22 Autres événements 443
19.23 Fonctions 446
19.24 Gestion des erreurs 446
19.25 Duplication de données d'un formulaire avec sous-formulaire 447
19.26 Calendriers 451
19.27 Identification 454
19.28 Utilisation de la sécurité avec VBA 458
19.29 Automation 461
19.29.1 Ouvrir MS Excel et y faire des traîtements 461
19.30 D.A.O. et ADO (avec MS Word ou MS Excel) 462
19.30.1 D.A.O. 462
19.30.1.1 Méthodes de lecture et écriture de tables et requêtes en DAO 462
19.30.1.2 Mise à jour de tables DAO 465
19.30.1.3 Recherche VBA DAO 466
19.30.1.4 Exécution requête d'action VBA DAO 467
19.30.2 A.D.O. 468
19.30.2.1 MS Excel - lecture ADO 470
19.30.2.2 Curseurs adLockReadOnly, adLockPessimistic, adLockOptimistic, adLockBatchOptimistic 470
19.30.2.3 MS Excel - Recherche ADO 472
19.30.2.4 MS Excel - Écriture ADO 472
19.30.2.5 MS Word - Recherche ADO 473
19.31 DLookup syntaxe 474
19.32 États 475
19.33 Requêtes 476
19.33.1 Lecture du contenu d'une Query DAO 477
19.33.2 ReQuery 477
19.34 Choix multiple 478
19.35 Calendrier 479
19.36 Connexion 481
19.37 Nouveautés VBA Access 2007-2010 482
19.37.1 Connecteur ADODB 482
19.37.2 Export flat-file (schema.ini) 483
19.37.3 Mode Hors-Ligne/En-Ligne SharePoint 483
20 Confusions courantes 484
20.1 Masques VS Valide Si, VBA & Null interdit 484
20.2 Valide Si VS VBA & Null interdit 485
20.3 Null interdit VS VBA 486
20.4 Intégrité VS Limité à la liste & VBA 486
20.5 Clés primaires combinées VS Macro Requery & VBA 487
20.6 État VS Formulaire A4 487
20.7 Sécurité MDW VS Sécurité VBA 487
21 VBScript 489

22 XML – XSL *490*
22.1 Import XML........ 493
23 Reverse Engineering (rétro-conception)........ *497*
24 A.S.P. et MDB *500*
24.1 Affichage de données........ 502
24.2 Ajout de données 503
24.3 Mise à jour d'enregistrements 504
25 P.H.P. et MDB........ *508*
26 MS Infopath........ *511*
26.1 Ajout de données 511
26.2 Echange de données........ 517
27 MS SQL Server........ *528*
27.1 Reverse engineering vers SQL server........ 528
27.2 Connexion à SQL server........ 535
28 AS/400 .... *539*
29 Oracle Express 11.2g........ *543*
30 MySQL........ *551*
31 PocketPC........ *556*
32 Business Objects........ *561*
33 Crystal Reports........ *567*
34 Runtime........ *569*
34.1 Activer le clic droit dans le runtime........ 570
35 Optimisation de bases de données *573*
35.1 Système FAT (File Allocation System)........ 574
35.2 Système ISAM (Indexed Sequential Access Method)........ 574
35.3 Arbres 575
35.3.1 Arbre B 576
35.3.2 Arbre B+........ 578
35.3.3 Arbre-R........ 580
36 Business Intelligence........ *581*
36.1 E.T.L. 581
36.2 Datawarehouse........ 582
36.3 OLAP........ 585
37 Limites MS Office Access........ *588*
37.1 Limites MS Access 2000+2002+2007+2010 588
38 Nouveautés MS Office Access *593*
38.1 Propositions de nouveautés ultérieures........ 595
39 Raccourcis claviers *596*

TABLE DES FIGURES

Figure 1Exemple MCD 22
Figure 2 Exemple MLD UML (ISO/IEC 19501) 22
Figure 3 Exemple MLD version "relationnelle" 22
Figure 4 Modèle Logique de Données (MLD) dans MS Visio 37
Figure 5 Accès BDD à distance (ADP) 47
Figure 6 Schéma (simple) de la BDD 109
Figure 7 Exemple de BDD (taille standard) 112
Figure 8 Requête d'union 174
Figure 9 Calendrier Access 208
Figure 10 Tableau croisé dynamique 214
Figure 11 Schéma SQL Server 534
Figure 12 Architecture type d'un ETL avec Datawarehouse 582
Figure 13 Ensemble des composants dans un système décisionnel (Source: 'Informatique Décisionnelle' NFE115) 584

TABLE DES TABLEAUX

Tableau 1 Type de bases de données 21
Tableau 2 Légendes Modélisation BDD 44
Tableau 3 Codes pour masque de saisie 66
Tableau 4 Codes de masque de saisie types 67
Tableau 5 Raccourcis clavier démarrage 301
Tableau 6 Stratégies de déploiement 350
Tableau 7 Autorisations d'accès à une base 375
Tableau 8 Type de données VBA 399
Tableau 9 Objets VBA 405
Tableau 10 Tableau ASCII Standard 406
Tableau 11 Tableau ASCII étendu 406
Tableau 12 Objets MS Access 407
Tableau 13 Objets MS Access en mode ouverture 407

1 Liens internet

Voici une liste non exhaustive de quelques excellents liens concernant MS Access!

À visiter absolument !!!

http://access-templates.com/ (des centaines de modèles de BDD Access déjà faites)

http://ww.testoffice.com (testez vos connaissances)

http://msdn.microsoft.com (page des développeurs)

http://msdn.microsoft.com/fr-fr/library/bb726434%28v=office.12%29.aspx (lien direct VBA)

http://communities.microsoft.com (newsgroups)

http://mypage.bluewin.ch/w.stucki/astuces.htm (page perso)

http://ww.info-3000.com/access/supportdecours/index.htm (page perso)

http://access.seneque.free.fr (page perso)

http://ww.self-access.com/access (page perso)

http://memoaccess.free.fr (page perso)

http://cerig.efpg.inpg.fr/tutoriel/bases-de-donnees/sommaire.htm (page universitaire)

http://ww.wrox.com (livres MS Access en anglais)

http://ww.developpez.com (forum)

http://ww.microsoft.fr (pour les mises à jour et plug-ins)

http://w.mvps.org/accessfr/ (page des MVP)

http://homepage.bluewin.ch/wstucki/astuces.htm (plein d'astuces VBA!)

http://www.mvps.org/access/general/gen0012.htm (Leszynski/Reddick naming convention)

http://www.techonthenet.com/access/functions/index.php (liste de fonctions Access avec exemples)

http://www.bandwood.com/gbs.htm (plug-in permettant de créer rapidement des Gantt sous Microsoft Access: Gantt Chart Builder System)

http://www.iheartmacros.com (site officiel du VBA de Microsoft Office)

1. À propos du rédacteur



Nom Prénom: ISOZ Vincent
Domicilié à ce jour à Lausanne (Suisse)

Formation: Ingénieur Physicien HES (B.Sc.)
Équivalence Internationale: Bachelor of Science

Année de naissance: 1978

Actuellement, je suis consultant trilingue (français, anglais et allemand) en mathématiques appliquées dans le tutorat d'analystes quantitatifs (niveau Bac+5 à Bac+7) et auteur de plusieurs livres électroniques dans les domaines suivants:

- maîtrise statistique des processus/procédés (méthodes paramétriques et non paramétriques)
- modélisation prévisionnelle/décisionnelle avancée (arbres de décisions, chaînes de Markov)
- recherche opérationnelle (simplexe, algorithmes génétiques, algorithme GRG)
- data mining (réseaux de neurones, ACP, AFC, régressions, scoring, clustering, etc.)
- modélisation du risque en gestion de projets et finance d'entreprise (monte-carlo, etc.)
- gestion de projets (modèles et best practices théoriques EFQM+Six Sigma, MS Project)
- ISO 9001:2008, 5807:1985, 10015:1999, 31000+31010:2009, 8258:1991, 10017:2003, etc.
- Adobe Photoshop et Illustrator
- 12 applications de la suite Microsoft Office System (Project, Visio, SharePoint, Access, etc.)

À ce jour interventions dans plus de ~200 entreprises dont 10 du *Fortune 500* selon listing 2009 et 3 universités et écoles d'ingénieurs suisses dans des cours de modélisation de bases de données et simulations stochastiques du risque. Formation de plusieurs dirigeants de multinationales en one to one.

Accessoirement j'interviens pour des formations sur des logiciels comme MS Project, MS Visio, MS Access et une vingtaine d'autres dont je délègue l'organisation à des entreprises spécialisées dans la formation continue en bureautique (niveau licence et en-dessous).

2 Introduction

"*Je gère ma base de données sur Excel…*"

"*Je travaille sur une base de données Excel…*"

Il est encore fréquent d'entendre cette phrase, **un non-sens** puisque MS Office Excel à ce jour (2003) n'est pas un système de gestion de base de données mais un tableur!

Cette confusion trouve son origine en partie dans le fait que les interfaces des deux logiciels sont similaires, lorsque les données sont présentées en tableau. Mais la ressemblance s'arrête là! Le tableau se caractérise par sa souplesse et sa rapidité de mise en oeuvre, alors que le système de gestion de bases de données permet avant tout d'assurer la cohérence d'une grande quantité d'informations, indépendamment de leur présentation: tables, relations, intégrité référentielle sont autant de puissants outils spécifiques aux bases de données permettant d'assurer un contrôle permanent de la cohérence des informations qu'elles renferment.

Une base de données relationnelle (S.G.B.D.R.) a plusieurs avantages comparés à une base de données simple aussi appelée "flat file". Elle utilise beaucoup moins d'espace mémoire, car elle réduit au minimum les redondances ou les répétitions des données (si elle a été bien conçue).

Remarques:

R1. En prévision de l'évolution technologique des bases de données, il est habituellement préférable de mettre les tables dans un fichier MS Access prédéfini "indépendant" d'un autre fichier où se trouveront les formulaires, requêtes et états (voir page 350).

R2. Contrairement aux exercices donnés dans ce support, les noms des champs de données doivent être des mots uniques (sans espaces), en minuscules, sans accents et autres symboles spéciaux (c'est ce qui est fait pendant le cours).

R3. Pour plus d'informations sur les conventions de nommage voir la page 44.

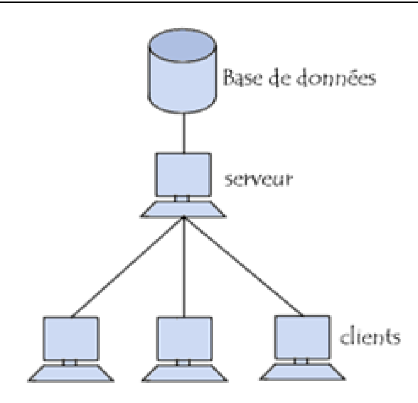
De plus, pendant le cours, nous n'allons pas nécessairement suivre l'ordre et la structure des exercices tels que présentés dans ce support (il faut savoir être flexible!).

Par ailleurs, en toute rigueur nous ne devrions pas parler de "base de données MS Access" mais de "base de données JET", JET (Joint Engine Technology) étant le moteur de base de données de MS Access 2003 et antérieur qui est lui-même un outil RAD (Rapid Application Development).

Remarque: MS Access 2007 et ultérieur utilisent un nouveau moteur appelé "Access Database Engine"…

Enfin, la stabilité des fichiers MS Access (fichiers *.mdb) n'est pas garantie. Il arrive qu'un fichier soit endommagé et, malgré les outils de réparation, ne puisse être récupéré. Il est donc impératif de soigner tout particulièrement sa politique de sauvegarde.

Pour finir, je pense sincèrement que les SGBDR clients sont des outils morts et désuets face aux bases de données internet (client-serveur):



qui permettent une beaucoup plus grande flexibilité, une interface plus conviviale, des possibilités grandioses, un environnement multi-utilisateur réel et j'en passe... Donc, réfléchissez sérieusement avant de faire une base de données avec des techniques qui ont 30 d'âge (est-ce que vous rouleriez avec une voiture dont la carrosserie est neuve, mais dont le moteur a 30 ans... probablement pas!).

2.1 Environnement

Certes, MS Access est un L4G[1] créé en 1992. Mais ce n'est pas le seul logiciel client de ce type sur le marché et pas le plus vieux! Je souhaiterais donc indiquer les grands classiques dans l'ordre de popularité afin d'être neutre:



Microsoft Office Access – Payant (www.microsoft.com)



FileMaker – Payant (www.filemaker.com)

[1] un L4G (sigle de langage/logiciel de quatrième génération) est un terme utilisé pour désigner un type de logiciel qui comporte un langage de programmation combiné avec un système de gestion de base de données.



Open Office Base – Gratuit (http://fr.openoffice.org)



4D – Payant (www.4d.com)

et nous pourrions encore citer:

- Borland Paradox
- IBM DB2
- Ingres
- Interbase
- Microsoft SQL server
- Microsoft FoxPro
- Hyperion EssBase
- Oracle
- H2
- Sybase
- MySQL
- MariaDB
- PostgreSQL
- Informix
- mSQL
- SQL Server
- Hadoop
- Apache Derby
- JavaDB
- MaxDB SAP
- SQLite
- Teradata

Pour plus d'informations:

https://en.wikipedia.org/wiki/List_of_relational_database_management_systems

2.2 Protocole

Lorsque l'on crée une base MS Access, on procède toujours de la façon suivante (les temps ci-dessous sont donnés relativement à une qualité professionnelle assurée - ces temps ne sont pas valables pour des amateurs où les facteurs peuvent aller de 0.1 à 100……:

1. Analyse des besoins (2 jours à 2 semaines)
2. Choix d'une stratégie de nommage[2] (1 à 2 heures)
3. Modélisation selon les formes normales (2 jours à 2 semaines)
4. Définir un *.mdw selon le protocole donné dans le chapitre de gestion de la sécurité voir page 354)
5. Création des tables (1 jour à 2 semaines)
    a. Définition des propriétés des tables[3] (types de données, formats, contraintes, légendes, index).
    b. Création des liaisons (après la création des listes de choix !!!) entre les tables (un à un, un à plusieurs, intégrité…)
6. Création des requêtes (simple, ajout, suppression, table…) (1 à 2 semaines)
7. Création et design des formulaires (simples, imbriqués – ne pas oublier les infobulles!) (1 à 2 semaine)
8. Création et design des états (1 semaine)
9. Séparation des tables dans un fichier indépendant: stratégie dorsale/frontale (1 jour)
10. Analyse/Optimisation de la base (1 jour)
11. Définir la gestion des utilisateurs (1 à 3 jours)
12. Compression de la base (1 à 5 minutes)
13. "Beta Testing" de la base (plusieurs jours) et ensuite validation (1 à plusieurs semaines)
14. Crypter la base (*.mde/*.accde) et la déployer en runtime

[2] Ne pas prendre comme exemple ce support de cours qui ne respecte pas à 100% la norme pour faciliter la compréhension du lecteur

[3] Ne pas oublier de mettre une clé primaire sur des champs AutoNum sur chaque table même si pas nécessaire dans un premier temps sinon problèmes en perspective lors de migrations !!!

Remarques:

R1. Dans une démarche rigoureuse, aucune donnée ne devrait être saisie dans les tables tant que la base n'est pas totalement terminée.

R2. Commentez vos tables, vos champs etc… et il vaut mieux en mettre trop que pas assez !!!

R3. Il est possible que dans certains cas l'ensemble de la base doive être faite en programmation V.B.A à cause de contraintes ou complications trop élevées.

2.3 MS Excel (tableur) VS MS Access (SGBDR)

L'expérience nous montre que plus de 95% des utilisateurs de la suite MS Office utilisent MS Excel à tort pour la gestion de listes et consolidation de données. Ce qui est incroyable c'est que ces utilisateurs se rendent pourtant bien compte que ce logiciel n'est pas fait pour la gestion de données étant donné les difficultés qu'ils rencontrent et la perte de temps que cela engendre!

Le problème vient au fait, d'un manque de recul et de formation de ses utilisateurs car rappelons-le, MS Excel est un TABLEUR (graphiques et calculs) et MS Access un système gestion de BASE DE DONNÉES RELATIONNELLES (cela veut bien dire ce que cela veut dire !!).

Certes, il est possible d'argumenter le fait que MS Access est un logiciel métier pour spécialistes (informaticiens) et que contrairement à MS Excel qui est étudié dès le gymnase (collège) dans les écoles publiques, MS Access n'est lui souvent seulement enseigné qu'en premier ou deuxième cycle universitaire dans les branches d'économie.

Cependant, toute entreprise doit se poser les questions suivantes:

1. Est-ce normal que les employés perdent des centaines d'heures par année dans MS Excel à faire ce qui peut être fait dans MS Access en 10 minutes (il est aberrant de faire de la consolidation, des TCD ou travailler à plusieurs dans un même fichier dans MS Excel ! On voit très bien que cela n'est pas fait pour !)

2. Sachant pertinemment que tout fichier MS Excel volumineux finit toujours un jour ou l'autre par se changer en une base MS Access, n'est-il pas plus avantageux de directement commencer par celle-ci et payer une formation d'une dizaine de jours à ses employés plutôt que de perdre du temps et de l'argent

3. Est-ce vraiment optimal que chaque employé fasse ses propres fichiers MS Excel, ou MS Access alors que dans une bonne entreprise, la majorité des données devraient être centralisées dans une unique base ?

4. Est-ce vraiment optimal que chaque employé enregistre ses documents sur son espace disque n'importe comment (alors qu'aujourd'hui des technologies comme MS SharePoint Portal Server permettent de faire de la gestion électronique de documents G.E.D. sur une base de données MS SQL Server)

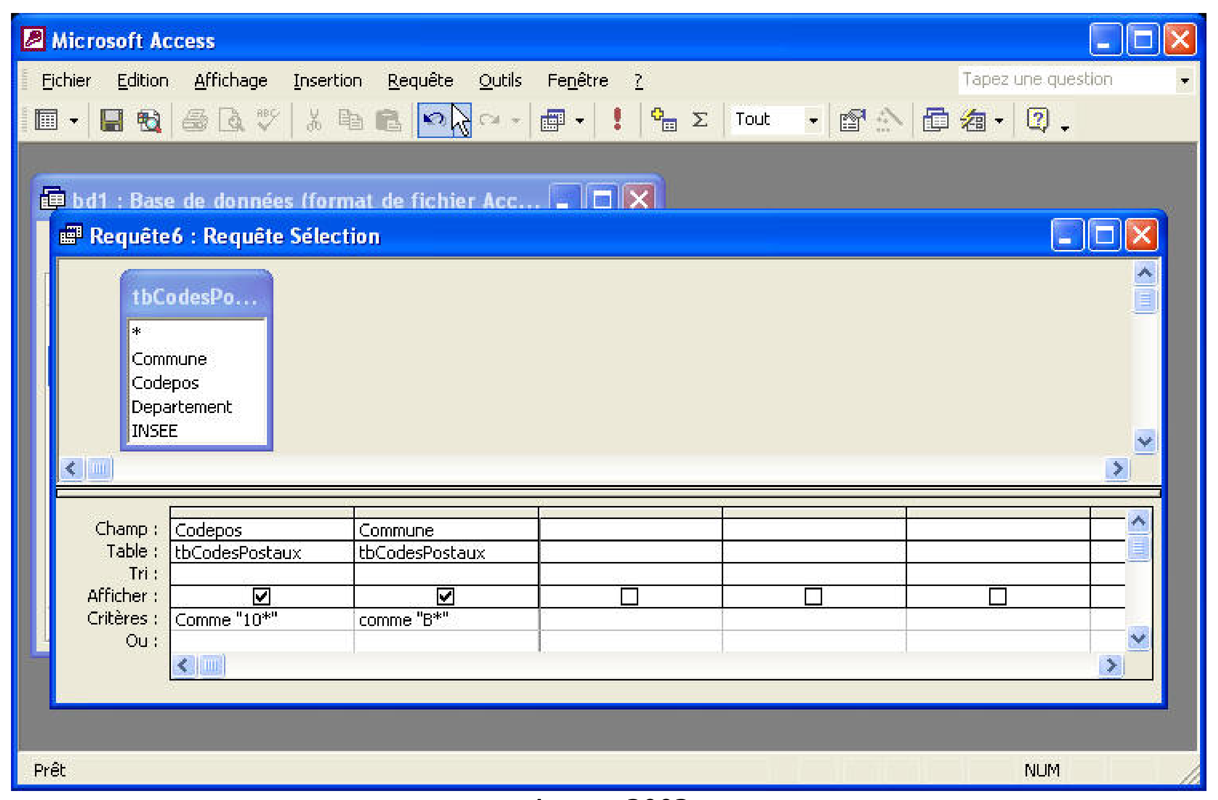
Ces observations proviennent de nos expériences dans plus de 400 PMI/PME et il nous paraît clair qu'il y a des sommes d'argent astronomiques perdues chaque année dans le suivi d'informations non organisées et par les mauvais usages de logiciels.

Il y aurait encore beaucoup à dire… comme:

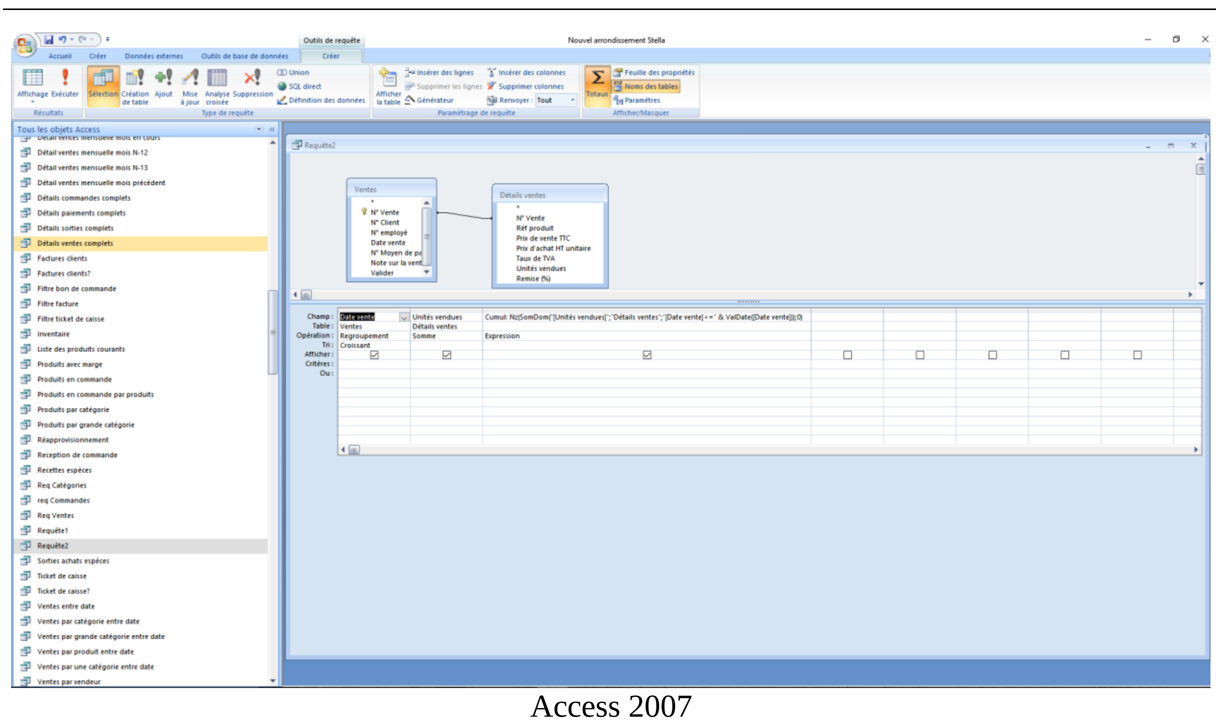
- MS Excel est lent pour l'analyse de millions de lignes de données
- MS Excel ne permet pas de faire un contrôle fin de la saisie sans programmation VBA
- MS Excel ne permet pas de faire des relations de un à plusieurs entre tables avec la fonction RECHERCHEV( ) ou sans passer par du VBA
- MS Excel n'est pas adapté à un environnement multi-utilisateurs
- etc.

Heureusement, les technologies à venir s'orientent toujours plus vers des systèmes GDD (gestion de données) ou GED (gestion électronique de documents) supportés par des bases de données relationnelles (comme SharePoint).

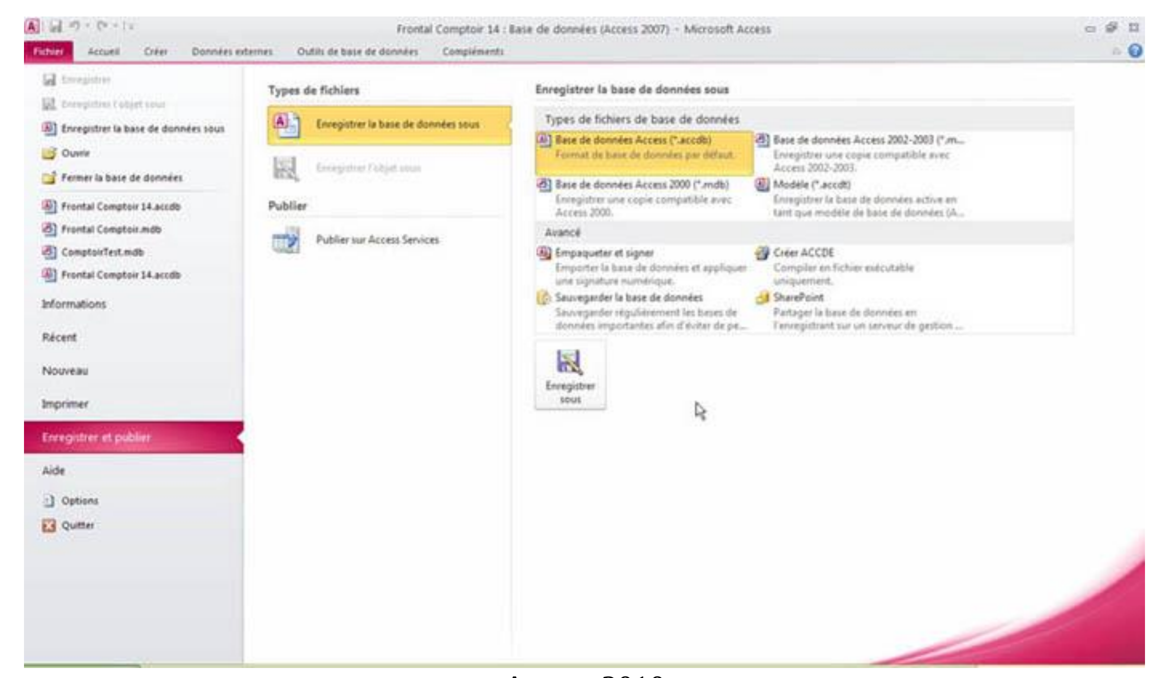
2.4 Évolution de l'interface



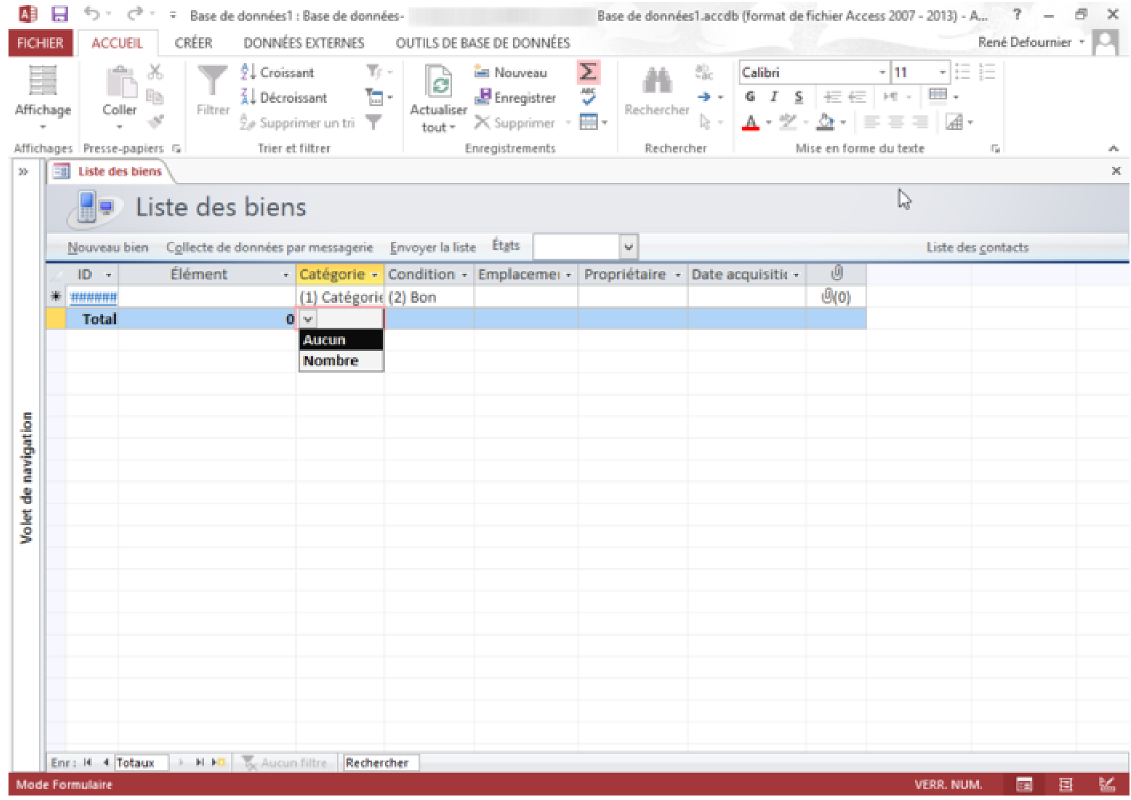
Access 2003



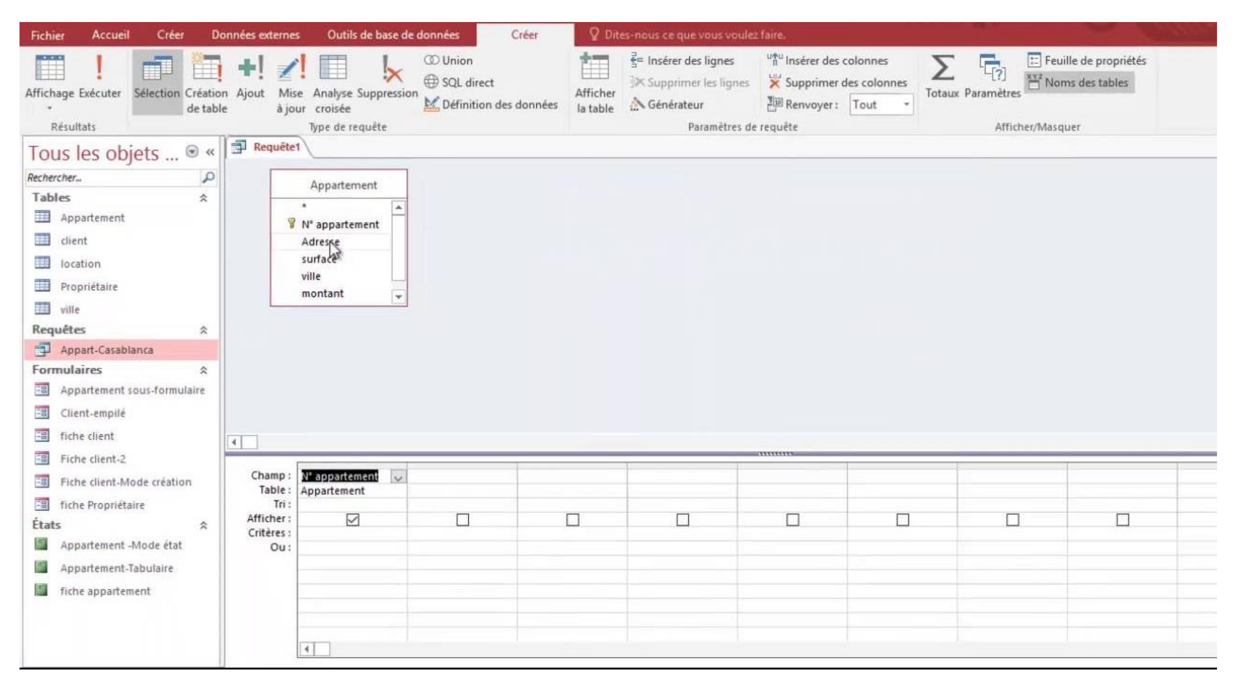
Access 2007



Access 2010



Access 2013



Access 2016

3 Notions de base de données

Ce chapitre traite des notions de fondamentales relatives aux bases de données et est largement indépendant du programme de base de données MS Access. Nous allons expliquer quelques concepts fondamentaux de la terminologie des bases de données et présenter certains outils et techniques de développement de base de données.

Lors du développement de processus de travail informatisés, il est fréquemment nécessaire d'enregistrer de gros volumes de données.

Un programme de base de données gère ces données et permet de les lire, de les modifier, de les supprimer et d'en ajouter. La façon dont les données sont enregistrées dépend du type de base de données et du système de base de données.

Définition:

En informatique, un **système base de données relationnelle (SGBDR)** est un stock d'informations décomposées et organisées dans des matrices appelées "tables" liées de manière directement ou indirectement connexes conformément au **modèle de données relationnel (MDR)**. Le contenu de la base de données peut ainsi être synthétisé par des opérations d'algèbre relationnelle telles que l'intersection, la jointure et le produit cartésien et est basé sur des algorithmes performants de gestion des données au niveau de la mémoire morte et de la mémoire vive de l'ordinateur.

Remarque: On parle généralement de système d'information (SI) pour désigner toute la structure regroupant les moyens mis en place pour pouvoir partager des données

3.1 Types de BDD

Il existe différents types de bases de données, les bases de données relationnelles étant les plus connues:

Type de BDD	Remarques
Base de données hiérarchiques	Il existe une hiérarchie entre les enregistrements (organigrammes). Un enregistrement subordonné appartient à un enregistrement supérieur (contrairement au modèle relationnel, le type des enregistrements d'une relation ne sont pas forcément identiques!). Nous parlons alors aussi de base de données en "étoile":

Base de données réseau	Ce modèle est semblable au modèle relationnel. Les relations entre les données sont toujours de type un à plusieurs et génèrent un graphique également appelé "réseau" ou "flocon": Les relations sont appelées des "ensembles" (cubes OLAP).
Base de données orientées objet	Les bases de données orientées objets ne comprennent pas de données, mais des objets (données + opérations sur les objets selon les méthodes de Programmation Orientée Objet). Par exemple SharePoint est basé sur une telle structure.
Base de données réparties	Les bases de données réparties enregistrent des données dans plusieurs bases de données (ex. sur des ordinateurs dans différentes villes) et possèdent des mécanismes de regroupement et d'interrogation des données réparties
Base de données relationnelles	Les systèmes de bases de données relationnelles (SGBDR) constituent le type de base de données le plus important. Les données sont enregistrées dans des tables liées par des relations. Le langage SQL a été défini pour l'évaluation des données

Tableau 1 Type de bases de données

3.2 Vues d'une base de données

Les divers groupes de personnes qui travaillent avec une base de données ont, selon leur tâche, des vues différentes de la base de données. Par exemple, le chef du personnel a besoin d'informations sur les employés, alors que l'administrateur de base de données s'occupe de l'enregistrement et de la sécurité des données. Le chef du personnel ne s'intéresse pas aux détails du projet de base de données physique. Ils sont définis et, au besoin, mis à jour par l'administrateur de base de données. L'architecture de base de données repose sur le projet logique, physique et visuel et l'ordre correspondant dans MS Access est souvent le suivant:

Vue interne (ou physique): La vue interne (ou "schéma interne") concerne l'organisation physique des données sur les disques. Les données peuvent être réparties sur plusieurs ordinateurs dans différents bâtiments et villes. Les données que vous affichez ultérieurement, par exemple, sous forme de table ne doivent pas nécessairement être enregistrées ensemble. Souvent la vue interne est associée à la terminologie de modèle physique de données (MPD).
Vue logique: Cette vue représente l'ensemble des données avec leurs relations sans souci d'implémentation physique ni d'utilisation. La conception de la base de données commence dans la vue logique: les données, leur gestion et leurs relations avec d'autres informations. Cette représentation peut être sous la forme d'un modèle conceptuel de donnée (MCD), soit une synoptique empirique mais compréhensible:

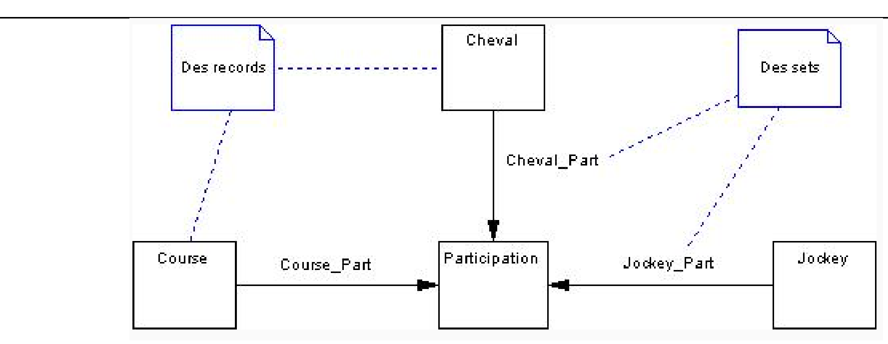


Figure 1Exemple MCD

Une fois le MCD effectué, nous passons normalement à un modèle logique de données (MLD), soit un synoptique respectant des règles bien précises comme Merise ou UML (ISO/IEC 19501):

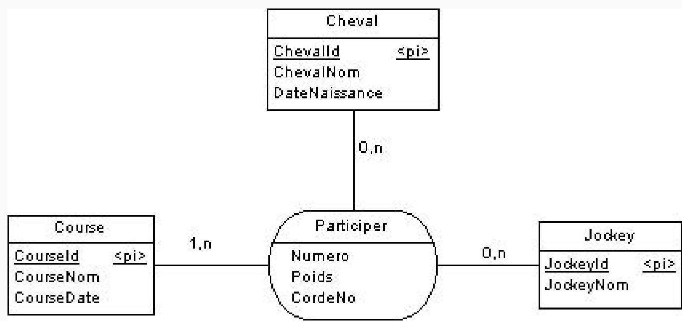


Figure 2 Exemple MLD UML (ISO/IEC 19501)

Ou en notation plus relationnelle:

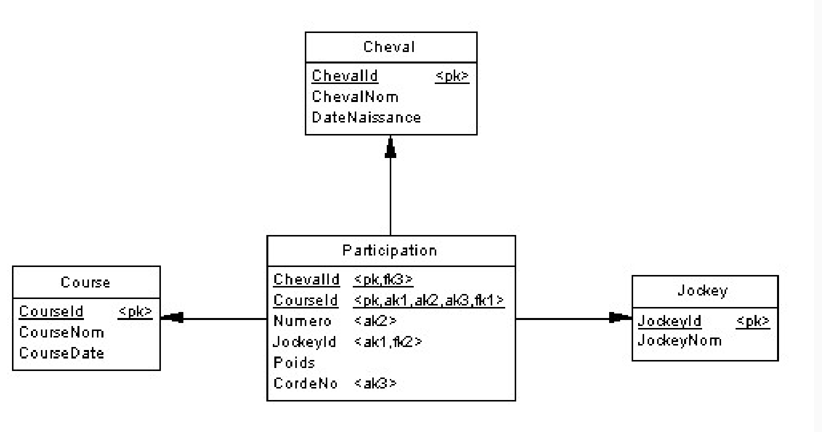


Figure 3 Exemple MLD version "relationnelle"

Quand nous travaillons par exemple sur une base de données du type relationnelle, nous parlons alors de modèle logique de données relationnelles (MLDR).

Remarque: Suivant les auteurs et les praticiens la définition de MLD/MCD est inversée!

Vue externe (ou utilisateur): Elle représente la vue de l'utilisateur, qui ne travaille qu'avec une partie des données. Il ne connaît ni la structure interne ni la vue globale de la base de données, mais uniquement les données qui lui sont visibles. Ces données peuvent être liées et groupées d'une autre manière que dans la vue logique.

Les utilisateurs d'une base de données ne doivent normalement pas connaître l'organisation physique des données. La structure physique des données peut donc être modifiée, sans répercussion sur la conception de la base de données ou la vue de l'utilisateur. En particulier, les applications qui utilisent la base de données continuent à fonctionner sans problème.

Au niveau interne, vous pouvez utiliser différents formats de table (ex. MS Access, dBase, MS SQL Server ou Oracle). L'utilisateur ne doit pas connaître l'origine des données.

La structure des données d'une vue logique est spécialement adaptée aux besoins et aux autorisations de l'utilisateur. Par exemple, le chef du personnel peut afficher les salaires des employés, alors qu'un collaborateur du département personnel ne peut accéder qu'à leurs coordonnées.

Projet de base de données logique	Le projet logique décrit la structure des données et les liaisons entre elles: ▪ rassemblement des données (ex. dans des tables) ▪ apparence des relations entre les données (ex. intégrité référentielle) ▪ ajout de la couche *Business Logic*: vérification de validité, formats de saisie, etc. Cette logique est alors à la disposition de tous les utilisateurs des données. Elle garantit une gestion cohérente des données.
Projet de base de données physique	Le projet physique porte sur l'enregistrement des données: ▪ répartition des données sur le disque ▪ possibilités d'accès aux données (réseau, internet, local) ▪ optimisation de l'enregistrement des données pour les opérations de recherche fréquentes ▪ intégration de mécanismes de sécurité automatiques (ex. enregistrement de données en plusieurs exemplaires, en miroir – en cas de défaillance d'un ordinateur, une autre machine peut prendre le relais avec les mêmes données)

L'architecture à 3 niveaux définie par l'architecture ANSI-SPARC[4] permet d'avoir une indépendance entre les données et les traitements. D'une manière générale un SGBD doit avoir les caractéristiques suivantes :

- le niveau physique peut être modifié indépendamment du niveau conceptuel. Cela signifie que tous les aspects matériels de la base de données n'apparaissent pas pour l'utilisateur, il s'agit simplement d'une structure transparente de représentation des informations.
- le niveau conceptuel doit pouvoir être modifié sans remettre en cause le niveau physique, c'est-à-dire que l'administrateur de la base doit pouvoir la faire évoluer sans que cela gêne les utilisateurs.
- le SGBD doit pouvoir éviter dans la mesure du possible des informations et manipulations qui seraient typiquement redondantes avec un tableur, afin d'éviter d'une part un gaspillage d'espace mémoire mais aussi des erreurs

[4] American National Standards Institute, Standards Planning And Requirements Committee

- des personnes ne connaissant pas la base de données doivent être capables de décrire leur requête sans faire référence à des éléments techniques de la base de données.
- le système doit pouvoir fournir les réponses aux requêtes le plus rapidement possible, cela implique des algorithmes de recherche rapides.
- les données doivent être cohérentes entre elles, de plus lorsque des éléments font référence à d'autres, ces derniers doivent être présents.
- le SGBD doit permettre l'accès simultané à la base de données par plusieurs utilisateurs.
- le SGBD doit présenter des mécanismes permettant de gérer les droits d'accès aux données selon les utilisateurs.

3.3 Modules d'un SGBDR

Un SGBDR client ou serveur est basé sur plusieurs couches pour fonctionner. Sans aller pour l'instant dans les détails en voici une énumération dans l'ordre du plus bas niveau de la couchée électronique jusqu'au niveau utilisateur que le lecteur comprendra normalement une fois lu l'ensemble du présent livre électronique:

- **Gestionnaire de fichiers (SGF)**
  Le gestionnaire de fichiers gère les affectations de mémoire sur disque et les structures de données qui représentent l'information sur disque. Il contient la description de l'organisation des fichiers (pages) de la base de données.

- **Gestionnaire de buffer**
  Le gestionnaire de buffer supervise les échanges entre les disques et la mémoire centrale grâce à une mémoire tampon.

- **Gestionnaire de la base de données**
  Le gestionnaire de la base de données sert d'interface entre les données au niveau physique et les applications visuelles.

- **Processeur de consultation**
  Le processeur de consultation transcrit les requêtes de consultation dans la mémoire physique en instructions compréhensibles par le gestionnaire de la base.

- **Optimiseur**
  L'optimisateur tente de formuler la requête de l'utilisateur de façon optimale en vue d'en accélérer l'exécution à partir de statistiques d'utilisation de la base de données.

- **Pré-compilateur DML**
  Le pré-compilateur DML convertit les instructions DML (Data Manipulation Langage comme SQL) du programme d'application SQL en procédures et codes adéquats pour être comprises par l'optimiseur.

- **Compilateur DDL**
  Le compilateur DDL convertit les instructions DDL (Data Definition Language comme SQL aussi) en un jeu de tableaux (pages) stockés dans un dictionnaire de données.

- **Gestionnaire d'accès et d'intégrité**
  Le gestionnaire d'accès et d'intégrité conserve l'intégrité des données et gère les habilitations des utilisateurs (autorisations d'accès).

- **Contrôleur multitâche**
  Le contrôleur multitâche traite les conflits éventuels entre traitements simultanés.

- **Module de récupération**
  Le module de récupération assure la cohérence de la base même à la suite d'un "crash".

- **Fichiers de données**
  Les fichiers de données contiennent la description de la base de données.

- **Fichiers systèmes**
  Le fichier systèmes conservent les données relatives à la structure de la base, les autorisations d'accès et le dictionnaire des données.

- **Index**
  Les index assurent la rapidité d'accès aux données de la base au niveau des requêtes (SQL).

- **Statistiques d'utilisation de la base**
  Les statistiques d'utilisation de la base sont exploitées par le module optimiseur.

3.4 Méthodes de création de BDD

Pour la gestion de données, vous avez besoin d'un logiciel appelé "système de gestion de base de données" (SGBD). Il est principalement chargé de traiter les requêtes des utilisateurs et de renvoyer les données désirées. Un SGBD utilise les trois vues d'une base de données. Par exemple, si un utilisateur tente de demander des données sur un client, le SGBD peut exécuter les opérations suivantes:

- La requête est convertie en procédure interne afin de préparer l'enregistrement désiré

- La vue utilisateur est convertie en vue logique et les structures de données nécessaires sont recherchées.

- L'emplacement d'enregistrement physique des données est défini.

- L'accès aux données s'effectue avec des opérations du système d'exploitation

- Les données sont transmises à l'utilisateur

Le SGBD est donc à l'origine de plusieurs transformations des vues. En outre, il peut tenir compte des autorisations de l'utilisateur, noter les modifications des données, effectuer des vérifications lors de la saisie de données ou les prendre en considération lors de la suppression de données et fonctionner en mode multiutilisateurs.

La création d'un projet de base de données n'est généralement pas une mince affaire comme nous l'avons vu dans notre protocole au début de ce support. Vous devez rassembler toutes les informations à gérer dans la base de données, réaliser différentes vues utilisateur, répartir

éventuellement les données sur plusieurs ordinateurs, etc. Vous devez choisir un modèle de données selon le système de base de données utilisé et le type de données, ainsi qu'une base de données prenant ce modèle en charge.

Si vous optez pour le modèle de données relationnel, les données sont enregistrées dans des tables. Vous pouvez définir des relations et des règles d'intégrité entre les tables.

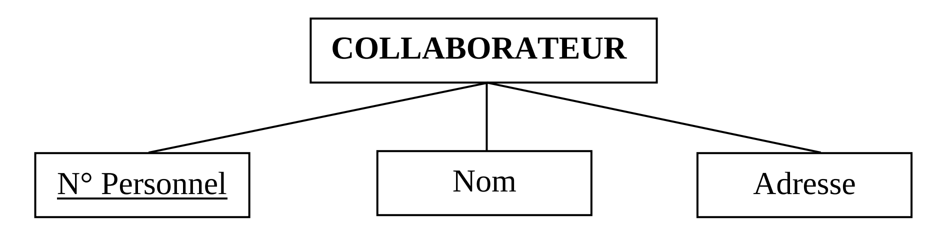
Pour éviter la répétition de données identiques dans différentes tables et une augmentation excessive du volume des informations enregistrées (erreurs nombreuses dans un enregistrement), vous pouvez soumettre les données à un processus de normalisation.

Il est également utile de posséder une vue d'ensemble des données sous la forme d'un graphique. Pour ce faire, vous pouvez utiliser le modèle E.R. (entité-relation).

Modèle de données relationnel: représentations des données dans des tables pouvant être liées.
Règles d'intégrité: Règles directement définies dans les tables et vérifiant l'intégrité des données en cas de modification, de suppression et d'ajout.
Processus de normalisation: méthode d'enregistrement sans redondance des données (pas d'enregistrement multiple) permettant d'éviter les erreurs lors de la suppression, de la modification et de l'insertion de données.
Modèle entité-relation/association: représentation graphique de toutes les informations importantes pour les tables définies et des relations entre elles.

Pour identifier de manière univoque tous les enregistrements d'une table, vous pouvez, par exemple, ajouter à cette dernière un attribut supplémentaire qui numérote les enregistrements. Il est alors possible distinguer les entrées de la table grâce à cet attribut. Dans ce cas, l'attribut est la "clé primaire" de la table. Chaque table ne peut posséder qu'une clé primaire qui permet d'identifier de manière unique chaque enregistrement et qui n'est pas susceptible de changer au cours du temps.

Dans l'exemple ci-dessous, l'ensemble d'entités *Collaborateurs* possède la clé primaire *N° Personnel*:



Remarque: comme nous le verrons dans les détails plus loin, la clé primaire d'une table peut être un attribut ou une association d'attribut dont la valeur permet d'identifier de manière unique les enregistrements de cette table.

Une "clé externe" est un champ d'une table qui crée une liaison avec un champ de clé primaire d'une autre table. Par exemple, tous les enregistrements d'une table *Commandes* contiennent le numéro du client qui a passé commande. Ce numéro identifie de manière univoque un client dans la table des clients. Le numéro de client dans la table des commandes constitue donc une clé externe ("clé étrangère" d'une autre table).

Il arrive fréquemment aussi que les enregistrements d'une table doivent être organisés dans un ordre différent (tris). Dans les grands volumes de données, cette procédure peut toutefois prendre du temps. Pour que l'accès aux données soit plus rapide, vous pouvez d'autres index que la clé primaire (nous verrons effectivement que la clé primaire crée automatiquement un index), appelés "index secondaires".

Remarque: Pour plus d'informations sur les relations voir la page 88

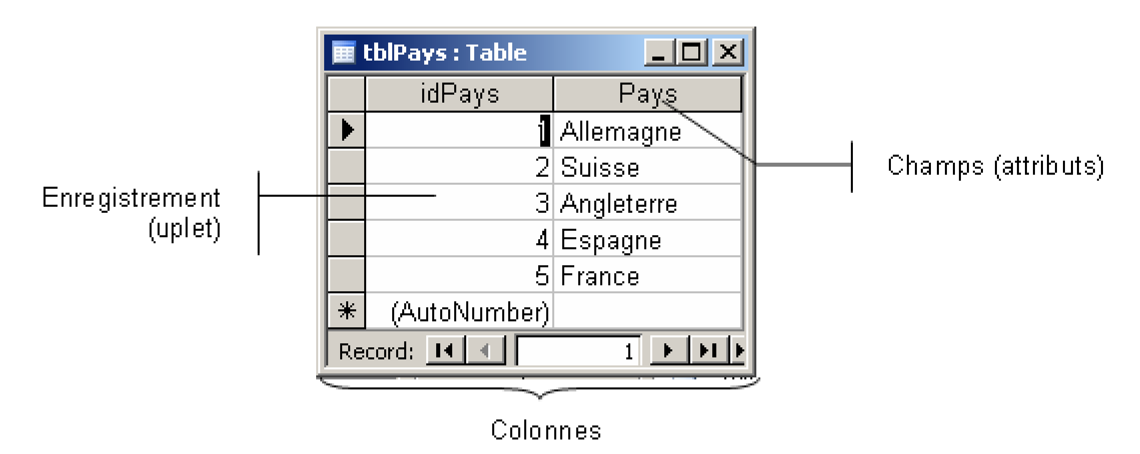
3.5 Modèle entité-relation (MER)

Le modèle de données relationnel a été développé en 1970 par le mathématicien E.F. Codd et écrit à l'aide de la théorie des ensembles. Ce modèle sert de base aux bases de données relationnelles.

Pour tenir compte de toutes les informations pertinentes dans un système de base de données, il est utile de décrire le domaine d'application et les objets nécessaires. L'outil graphique le plus connu et le plus utilisé à cet égard est le "**modèle entité-relation**" (MER) appelé aussi "**modèle entité-association**" (MEA) qui illustre tous les éléments d'un système de base de données relationnel et présente les rapports entre eux.

Dans la terminologie des bases de données, une "relation" est une "table" et donc une construction de colonnes et de lignes. Dans le domaine des bases de données, une relation ne désigne donc pas, par définition, une liaison.

Les données sont enregistrées dans une table. Elle représente les informations à gérer (ci-dessous une table typique de MS Access):



Une table est constituée de colonnes, également appelées "champs" ou "attributs". Les lignes de la table contiennent des "enregistrements" ou "uplets". Tous les enregistrements d'une table ont une structure identique.

La structure d'une table est également appelée son "schéma". Les divers systèmes de base de données (ex. MS Access, dBase ou Paradox) disposent de différents formats de table, dont dépendent les conventions de noms des tables et des champs et les types de données de champs.

Définitions:

D1. Une "entité" (ou "enregistrement" donc) peut être un objet, un concept ou une personne. Les entités se distinguent les unes des autres par leurs propriétés.

D2. Un ensemble d'entités est un rassemblement d'entités avec des propriétés identiques et correspond à une "table".

Remarque: lLrs de la création d'une base de données réfléchissez d'abord aux entités et donc aux ensembles d'entités qu'elle doit contenir.

D3. Une "relation" est une "liaison d'entités". Les relations se distinguent les unes des autres par leurs propriétés.

Le type de relation décrit les rapports numériques entre les différents éléments par exemple: combien d'enregistrements de la table "Collaborateur" ont été attribués à un enregistrement de la table "Département" ? Le type de relation est généralement défini par les mentions suivantes:

0 – Pas d'attribution
1 – Une seule attribution
n,m – Plusieurs attributions

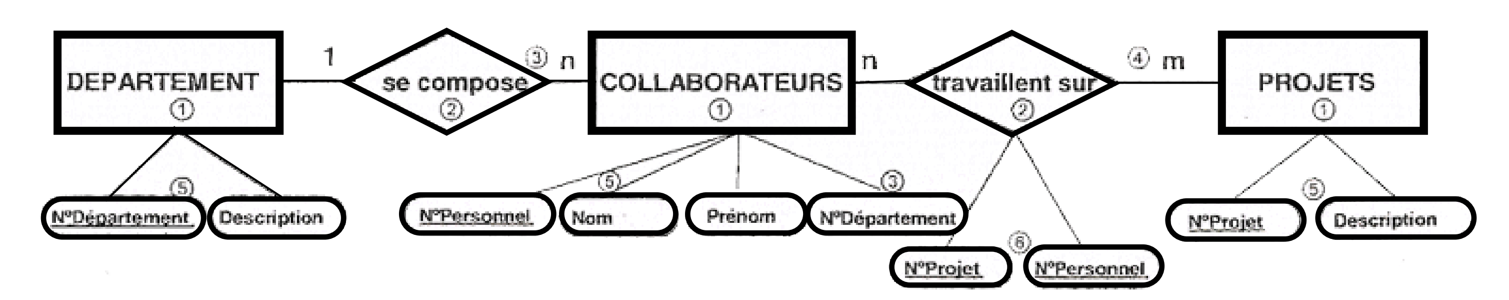
Les relations sont représentées par des lignes de connexion dans le modèle ER: la relation est décrite par un symbole et la "cardinalité" est ajoutée en fin de ligne.

D4. Les "propriétés ou "attributs" caractérisent une entité, un ensemble d'entités, une relation ou un ensemble de relations.

D5. Les propriétés sont données par les "noms des champs de données" d'une table.

D6. Un "domaine de définition" indique la plage de valeurs autorisées pour une propriété.

Exemple: une entreprise englobe des collaborateurs, des départements et des projets. Une base de données relationnelle doit être créée pour gérer toutes les informations et tous les événements. Dans l'illustration suivante, tous les éléments de la base de données et leurs relations sont rassemblés:



① Tous les résultats et informations peuvent être répartis en trois ensembles d'entités: *Département, Collaborateur* et *Projets*.

② Deux relations sont distinguées entre les ensembles d'entités: un département se compose de collaborateurs, qui travaillent sur des projets.

③ Étant donné qu'un département comprend plusieurs collaborateurs, mais qu'un collaborateur n'appartient qu'à un département, il s'agit d'une relation un à plusieurs. Dans ce cas, il est nécessaire d'insérer le champ de clé (primaire) de la table primaire (Département) dans la clé (étrangère) de la table étrangère (Collaborateurs) N° Département.

④ Plusieurs collaborateurs peuvent travailler sur plusieurs projets. Il s'agit d'une relation donc d'une relations plusieurs à plusieurs.

⑤ Les ensembles d'entités Département, Collaborateurs et Projets possèdent certaines propriétés. Par exemple, l'ensemble d'entités Projets possède les propriétés N° Projet et Description. La propriété N° Projet identifie exactement une entité ou un enregistrement et est donc choisie comme clé primaire.

⑥ Une relation plusieurs à plusieurs nécessite un autre ensemble d'entités, appelé "table de transition" car l'association entre les collaborateurs et les projets n'est pas identifiable. Le nouvel ensemble d'entités comprend au moins les champs de clé des deux ensembles d'entités qui composent la relation plusieurs-à-plusieurs.

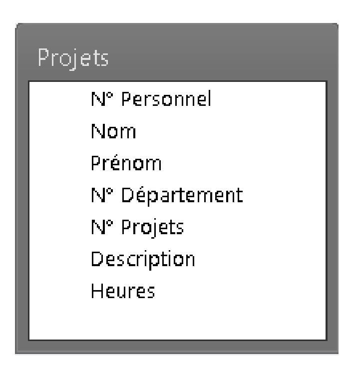
Pour obtenir une description compacte des entités, attributs et champs de clé, vous pouvez utiliser la forme textuelle suivante. Elle commence par le nom de l'entité (table), suivi, entre parenthèses, des attributs (champs), les champs de clé primaire étant soulignés.

DEPARTEMENT (N° Département, Description)
COLLABORATEURS (N° Personnel, Nom, Prénom, N° Département)
PROJETS (N° Projet, Description)
EVALUATION_PROJET (N° Projet, N° Personnel, HeuresTravail)

3.6 Normalisation

La répartition des données dans des tables relationnelles peut entraîner des erreurs lors de la modification, de l'insertion et de la suppression de données et ainsi générer des redondances et incohérences des données. Ces erreurs sont également appelées "anomalies".

Par exemple, chaque collaborateur est associé avec son département et ses données de projet selon la table ci-dessous:



Toutes les données sont enregistrées dans une seule table. Les problèmes suivants surviennent lors de la modification, de l'insertion et de la suppression de données:

N° Personnel	Nom	Prénom	N° Départemen	Designation	N° Projet	Description	Heures
1	Petous	Simone	1	Personnel	2	Promotion de v	83
2	Lamant	Marion	2	Achat	3	Analyse de la c	29
3	Demurs	Georges	1	Personnel	1,2,3	Enquête des cli	104,92,110
4	Colin	Jean-Pierre	3	Vente	2	Promotion	67
5	Bijoux	Janette	2	Achat	1	Enquête	160

1. Lorsque vous insérez un nouveau collaborateur qui n'a travaillé sur aucun projet, des champs de données sont laissés vides, ce qui gaspille de l'espace disque. Des problèmes de traitement peuvent également survenir en cas de requêtes, ex. lorsqu'un collaborateur travaille sur plusieurs projets en même temps. Ils figurent tous dans un champ *N° Projet* et doivent faire l'objet d'une opération supplémentaire. Une requête ne peut porter que sur tout le contenu d'un champ ou une partie de celui-ci.

2. Lorsque vous supprimez un collaborateur, vous devez également supprimer les données du projet correspondantes. Des données sont alors perdues.

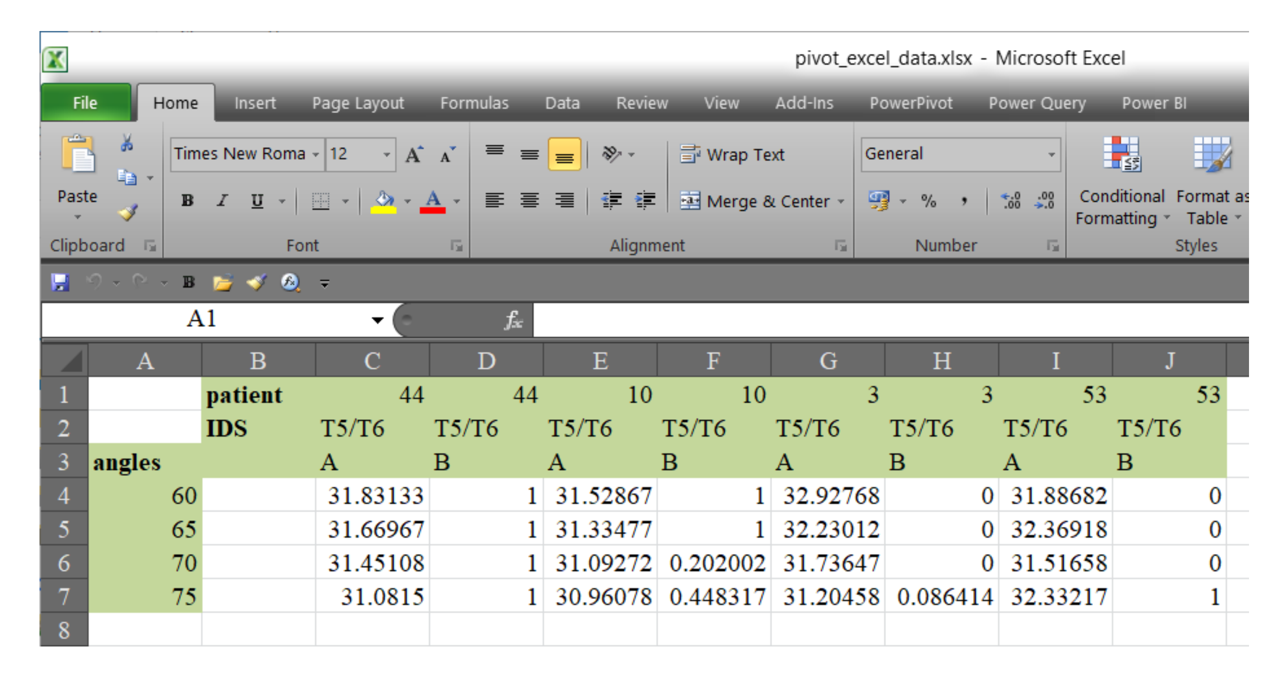
3. Lorsque la dénomination d'un projet a été modifiée, par exemple, *Enquête des clients* est remplacé par *Étude de marché*, tous les enregistrements contenant cette valeur doivent être modifiés (si plusieurs collaborateur y travaillent bien sûr !)

Le processus de normalisation permet notamment de résoudre les problèmes susmentionnés. Pour ce faire, les données des tables doivent respecter certaines règles. Le résultat de l'application de ces règles est appelé "forme normale" des tables.

Lors du processus de normalisation, les données sont réparties sur plusieurs tables. Les différentes étapes du processus de normalisation sont désignées sous le nom de formes normales: première, deuxième, troisième et quatrième formes normales.

La structure des données de l'exemple ci-dessus est appelée "non normalisée", car plusieurs informations sont enregistrées dans certains champs d'un enregistrement. Par exemple, le collaborateur *Demus* travaille sur les projets 1, 2 et 3.

Un autre exemple catastrophique et très courant de forme non normalisée dans les entreprises de la part d'employés n'ayant pas suivi de formation sur la gestion de données d'entreprise est la suivante:



	A	B	C	D	E	F	G	H	I	J
1		**patient**	44	44	10	10	3	3	53	53
2		**IDS**	T5/T6	T5/T6	T5/T6	T5/T6	T5/T6	T5/T6	T5/T6	T5/T6
3	**angles**		A	B	A	B	A	B	A	B
4	60		31.83133	1	31.52867	1	32.92768	0	31.88682	0
5	65		31.66967	1	31.33477	1	32.23012	0	32.36918	0
6	70		31.45108	1	31.09272	0.202002	31.73647	0	31.51658	0
7	75		31.0815	1	30.96078	0.448317	31.20458	0.086414	32.33217	1
8										

qui est bien évidemment elle aussi non normalisée mais en plus (ce qui est pire) sous forme dite de "pivot".

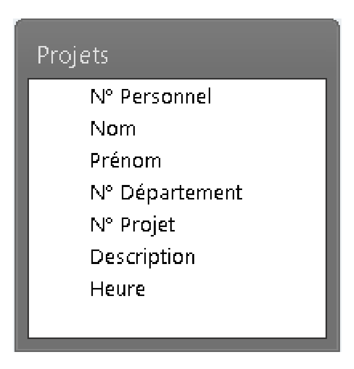
Je vais ici présenter ma définition personnelle des 4 formes normales (FN) de base (étant donné qu'il semblerait qu'il n'y ait pas de consensus absolu parmi tous les praticiens).

3.6.1 Première forme normale (1FN)

La première forme normale est la plus simple et souvent intuitive. Elle consiste à supprimer toutes les entrées multiples dans un champ. Chaque champ de données d'un enregistrement doit contenir une valeur maximum.

La première forme normale spécifie donc uniquement que chaque entrée d'un champ doit être indivisible (une valeur atomique soit: non composée) et en plus impose qu'il y ait au moins une clé primaire.

Cela revient donc à enlever le pluriel de certaines colonnes (seulement à *N° Projets* dans le cas présent):



Cela ne veut pas dire qu'une remarque sur un projet ne peut contenir qu'un mot. Si le champ contient toutefois plusieurs remarques avec des indications de date, l'entrée n'est plus automatique et peut être répartie en plusieurs champs de dates, avec textes de remarques correspondants.

N° Personnel	Nom	Prénom	N° Départemen	Designation	N° Projet	Description	Heures
1	Petous	Simone	1	Personnel	2	Promotion de v	83
2	Lamant	Marion	2	Achat	3	Analyse de la c	29
3	Demurs	Georges	1	Personnel	1	Enquête des cli	104
4	Colin	Jean-Pierre	3	Vente	2	Promotion	67
5	Bijoux	Janette	2	Achat	1	Enquête	160
6	Demurs	Georges	1	Personnel	2	Enquête des cli	92
7	Demurs	Georges	1	Personnel	3	Enquête des cli	110

Mais cette première forme normale a encore les problèmes suivants:

1. La table comprend des redondances. Des données de collaborateurs et des noms de départements et de projets apparaissent plusieurs fois.

2. La table contient les domaines indépendants suivants: collaborateurs, départements, projets.

3. Les données ne peuvent pas être identifiées de manière univoque. Par exemple, la description de certains projets ne peut être renvoyée qu'à l'aide d'un numéro personnel en plus.

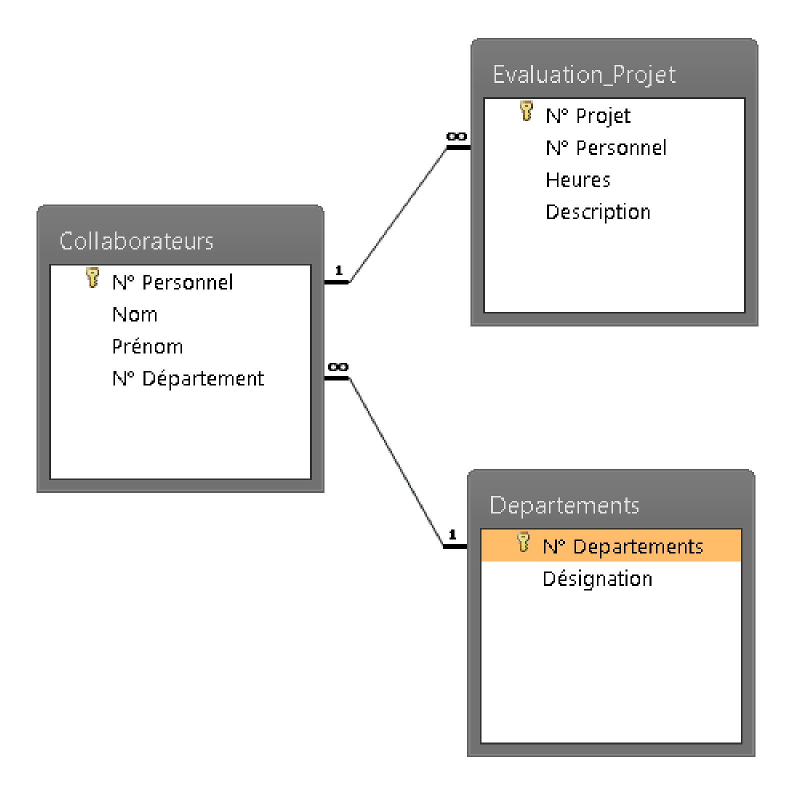
3.6.2 Deuxième forme normale (2FN)

La deuxième forme normale exige que chaque champ non-clé (étrangère) contenant des doublons dépende d'une clé (primaire) simple ou combinée.

Remarque: La deuxième forme normale appartient au modèle dit "modèle de normalisation de Boyce–Codd".

Les données de la table initiale sont désormais réparties dans trois tables:

COLLABORATEURS (N° Personnel, N° Département, Nom, Prénom)
DEPARTEMENT (N° Département, Désignation)
EVALUATION_PROJET (N° Projet, N° Personnel, Heures, Description)



3.6.3 Troisième forme normale (3FN)

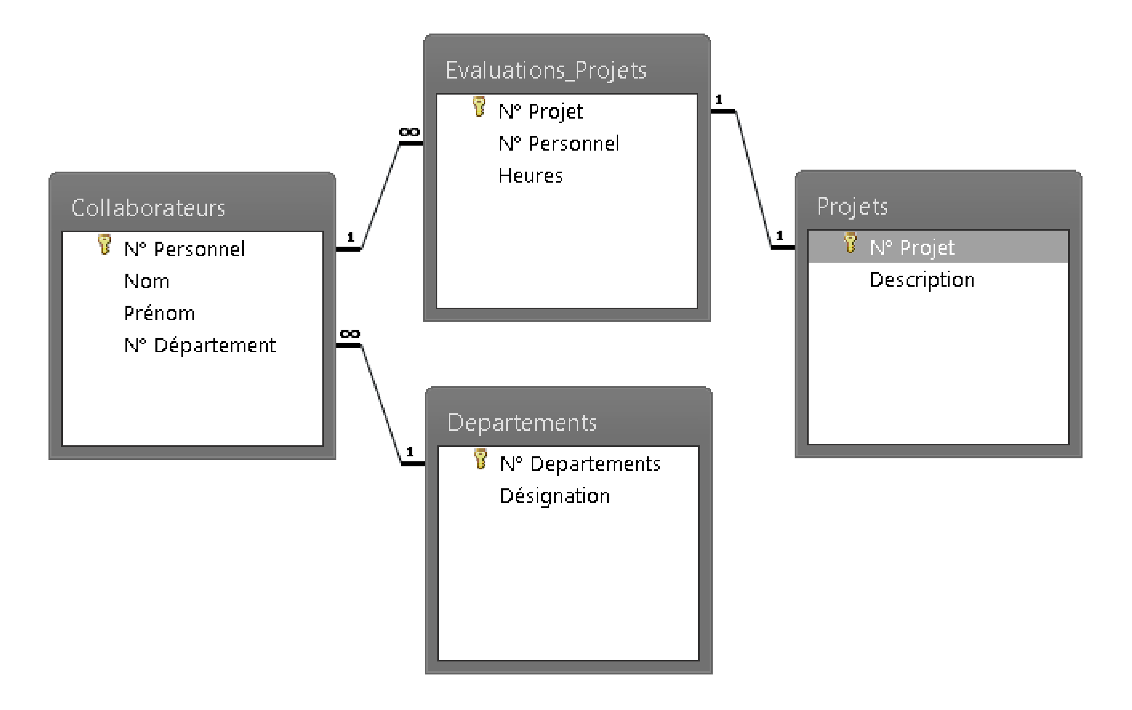
Une table répond à la troisième forme normale lorsque tous les champs de données ne dépendent que de clés primaires par l'intermédiaire de clés étrangères dans le sens d'une maximisation du découpage.

Les deuxième et troisième forms normales appartiennet au modèle dit "modèle de normalisation de Boyce–Codd".

Par exemple le champ *Description* n'a rien à faire dans *Evaluations_Projets* puisqu'elle se répète (ou peut se répéter!).

Un autre cas classique d'erreur de troisième forme normale est de trouver dans la table *Evaluations_Projets* des champs *Heures* et *Cout_Horaire* et de trouver la multiplication des deux dans la même table alors que les calculs ne doivent que rarement être stockés dans la pratique.

COLLABORATEURS (N° Personnel, N° Département, Nom, Prénom)
DEPARTEMENTS (N° Département, Désignation)
PROJETS (N° Projet, Description)
EVALUATIONS_PROJETS (N° Projet, N° Personnel, Heures, Description)



3.6.4 Quatrième forme normale

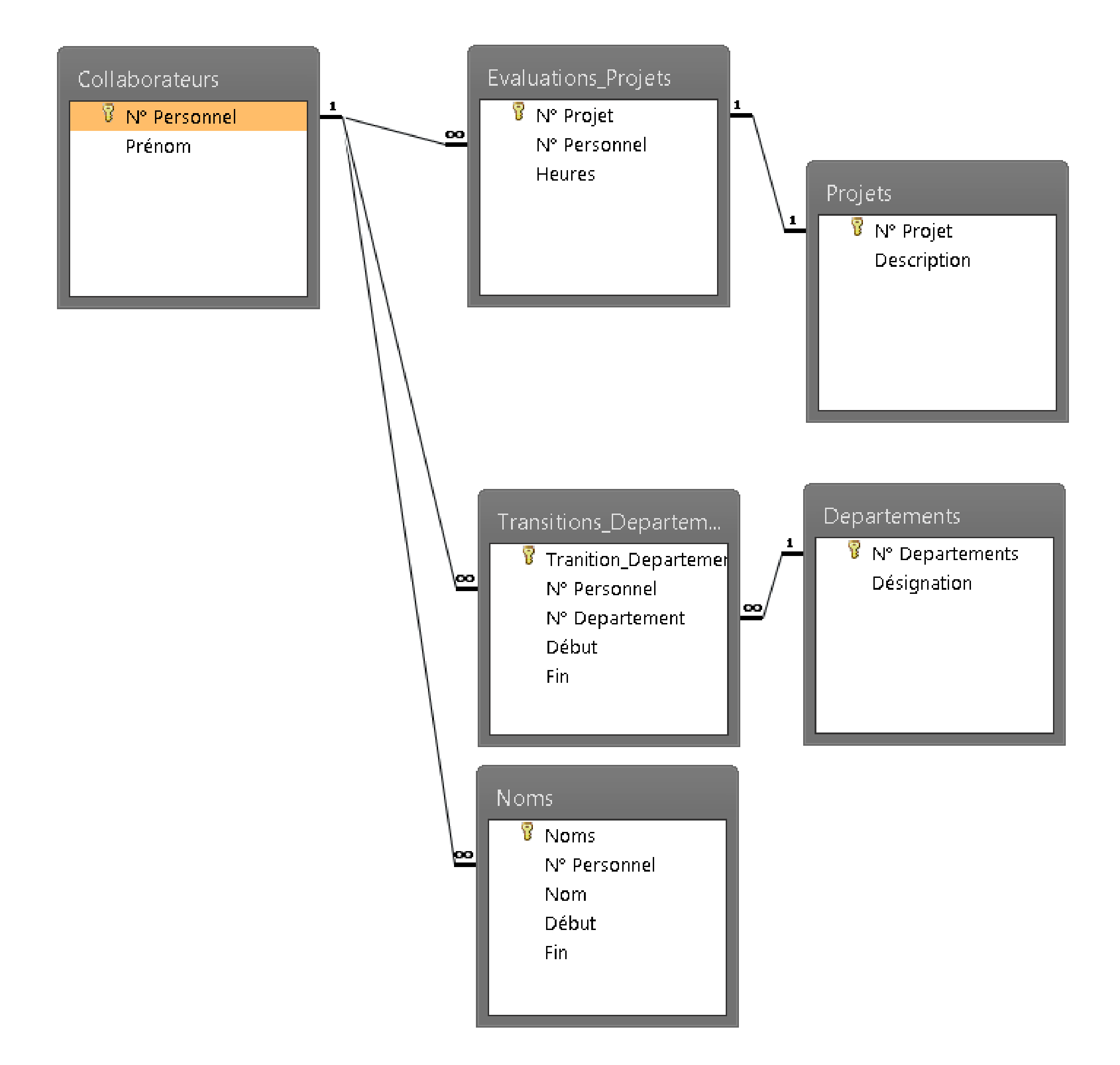
Pas tout les praticiens sont d'accord sur la définition de la quatrième forme normale donc voici la mienne:

La quatrième forme normale consiste à sortir les dépendances transitives cachées a priori dans des tables externes (toujours à l'aide d'une clé primaire et étrangère).

Par exemple le champ *Nom* à une dépendance transitive avec le *N° Personnel* car une collaboratrice peut très bien changer de nom par mariage. Il est donc mal venu de laisser le *Nom* là où il est si nous voulons plus avoir un historique utilisable de manière efficiente.

Par exemples le champ *Heures* à une dépendant transitive car rien ne nous dit qu'il va y avoir plusieurs estimations ou saisie du réel effectué et que dès lors il nous faut un change log (historique) des saisies.

Nous avons alors:



Remarque: Il y a même certains modèles avec une cinquième et sixième forme normale.

3.6.5 Exercice

Les données d'un magasin doivent être enregistrées dans une base de données. L'illustration suivante montre les données essentielles dans une table non normalisée. Esquissez un modèle ER pour les données indiquées:

Field Name	Data Type	
NomVendeur	Text	Nom du vendeur du magasin
PrenomVendeur	Text	Prénom du vendeur du magasin
RégionDeVente	Text	Région de vente du vendeur (Nord, Sud, Est, Ouest)
ArticleVendu	Text	Code de l'article vendu
NombreUnites	Number	Nombre d'unités vendues de l'article
TypePayement	Text	Type de méthode de payement (Visa, Cash,..)
DateVente	Date/Time	Date de vente
Fournisseur	Text	Nom des fournisseurs des articles
RueFournisseur	Text	Rue du fournisseur
NPAFournisseur	Text	Code postal fournisseur
VilleFournisseur	Text	VilleFounisseur
PaysFournisseur	Text	Pays du fournisseur
EntreeArticle	Text	Code article entrant dans les stocks
NombreUnites	Number	Nombre d'unités entrées dans les stocks
DateEntree	Date/Time	Date d'entrée dans les stocks de l'article
DesignationArticle	Text	Descriptif de l'article
MOQArticle	Number	Quantité minimal pouvant être commandée d'un article donné
PrixUnitéArticle	Number	Prix à l'unité d'un article

1. Normalisez le projet de base de données (jusqu'à la troisième forme normale).

2. Créez une base de données MS Access avec votre formateur soit directement dans le logiciel lui-même avec les outils à disposition soit en utilisant SQL, MS Visio ou MySQL Workbench (votre formateur vous aidera)

3. Quelles mesures pouvez-vous prendre pour garantir l'intégrité des données ?

4. Créez les relations nécessaires. Quelles relations doivent être créées avec l'intégrité référentielle et les mises à jour et suppression en cascade?

Remarque: N'hésitez pas à faire usage dans MS Access de l'outil *Analyse* (dans le menu *Outils*) qui va vous proposer de normaliser vos tables (voir les détails sur cet outil à la page 335)

3.7 Modélisation de BDD avec MS VISIO

Quand nous construisons directement les tables d'une base de données dans un logiciel de gestion des bases de données, nous sommes exposés à deux difficultés:

1. Nous ne savons pas toujours dans quelle table placer certaines colonnes (par exemple l'adresse de livraison se met dans la table des clients ou dans la table des commandes?).

2. Nous avons du mal à prévoir les tables de jonction intermédiaires (par exemple une table des articles vendus qui est indispensable entre les tables des clients et la table des articles)

Il est donc nécessaire de recourir à une étape préliminaire de conception et il s'agit d'une partie importante du travail du développeur est la modélisation de la base de données (sinon le

coût de correction est très important!). Pour exercer cela correctement (sans passer par une feuille A0 collée sur un mur et gribouillée de part en part), il existe des logiciels spéciaux tels que Merise ou MS Visio (pour ne citer que ceux que l'auteur connaît relativement bien).

Remarque: Il existe des cours de modélisation sur 2 jours, mais le sujet est au fait beaucoup plus étendu et fait l'objet d'un métier à lui seul. Le spécialiste dans le domaine est souvent informaticien théoricien ou mathématicien avec un excellent bagage d'algèbre relationnelle. De plus je conseille aux personnes travaillant avec des PME ou des consultants indépendants de travailler plutôt avec le logiciel gratuit MySQL Workbench pour la modélisation plutôt que MS Visio qui est très coûteur pour un indépendant ou une PME.

Nous allons nous restreindre dans le cadre de ce cours à l'utilisation de MS Visio 2000. La démarche a changé dans la 2002 et la 2003 et le forward-engineering (le fait d'exporter le schéma d'une base de données existante dans MS Visio) n'est disponible depuis la version 2007 que dans la version Vision Enterprise Architect (il y a quatre versions de Microsoft Office Visio à ma connaissance). Par contre, le reverse-engineering (le fait d'importer le schéma d'une base de données existante dans MS Visio) est toujours disponible dans la version pro et premium.

L'utilisation en étant très simple, le formateur va vous orienter sur la manière de procéder pour créer par exemple les schémas de base de données auxquels doivent arriver les participants à un cours MS Access après 3 à 4 jours de cours. Le schéma de notre base de données pour l'ensemble de la formation ressemblera au tout début au modèle logique de données (MLD) présenté ci-dessous.

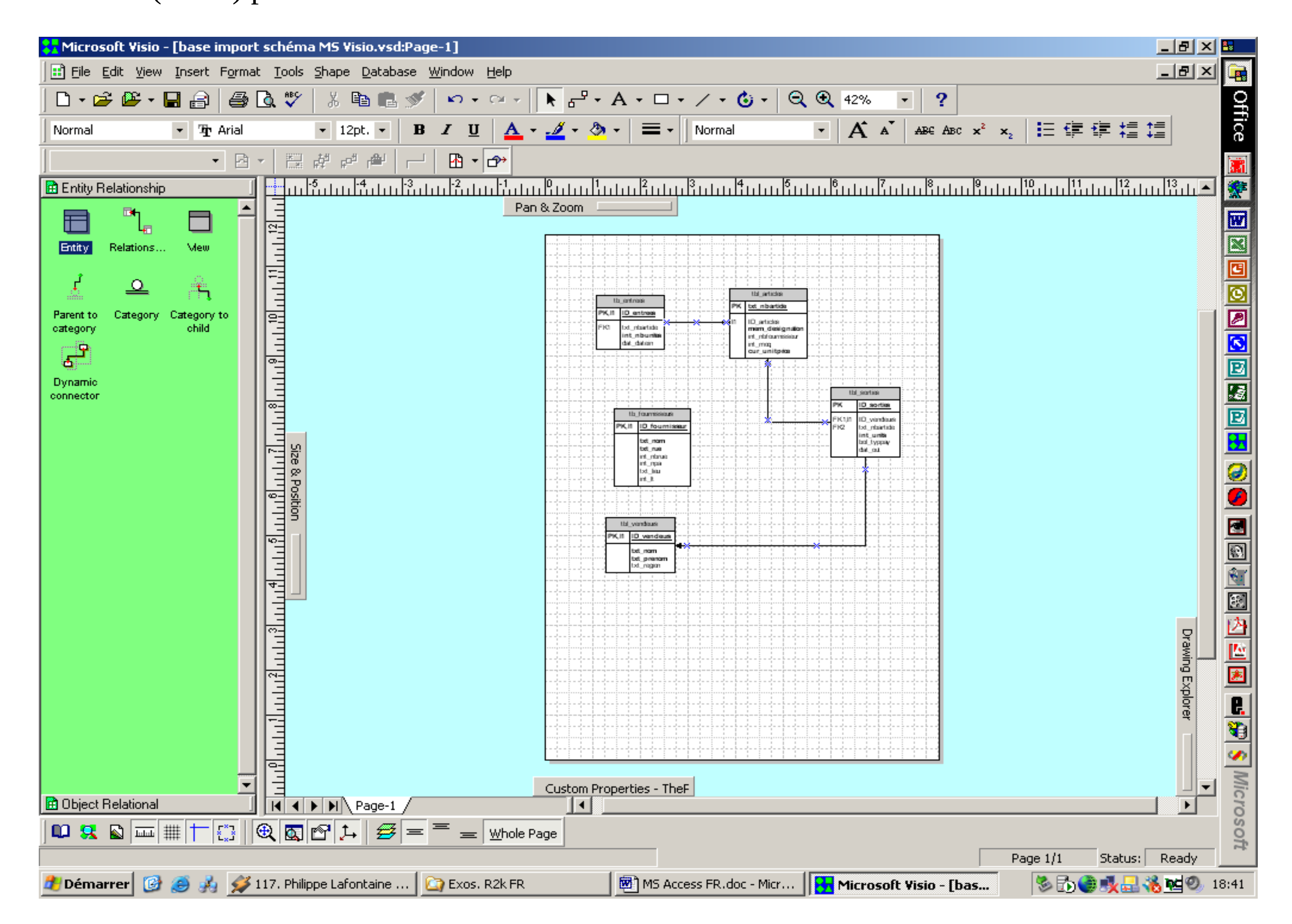
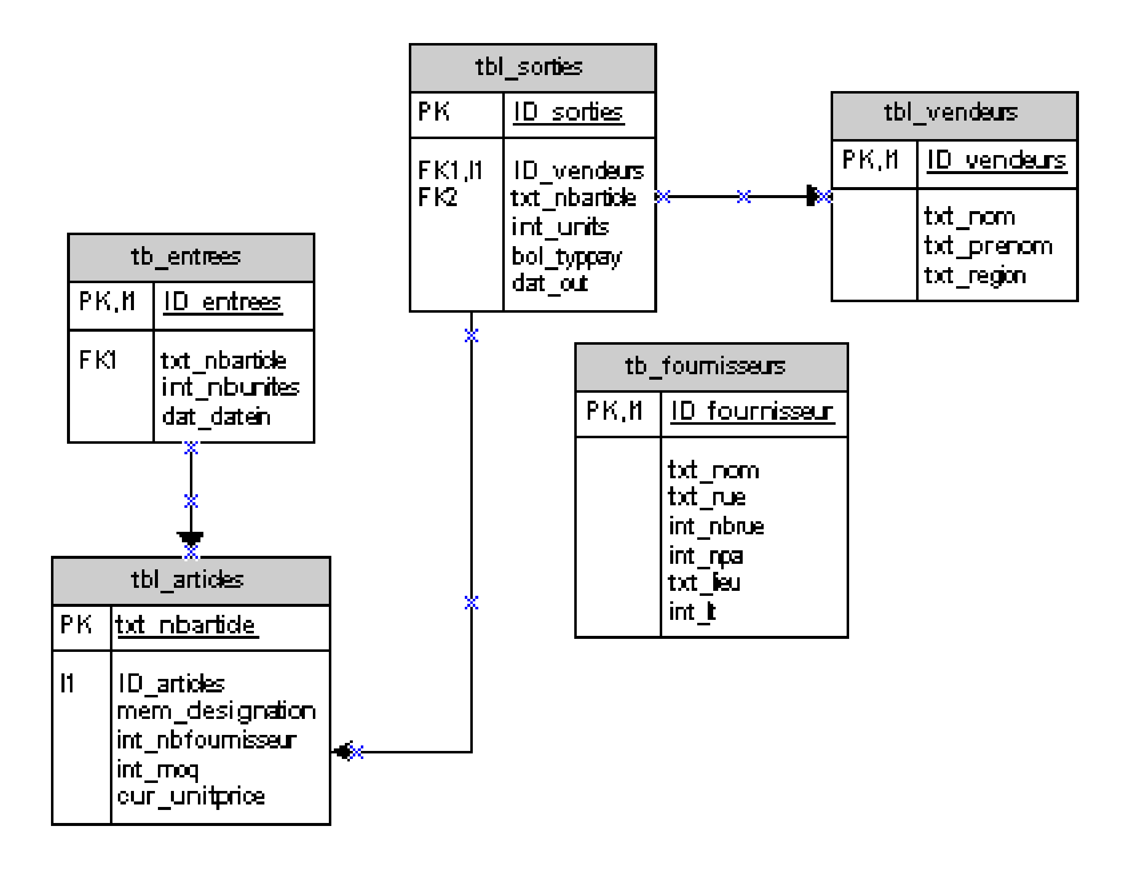


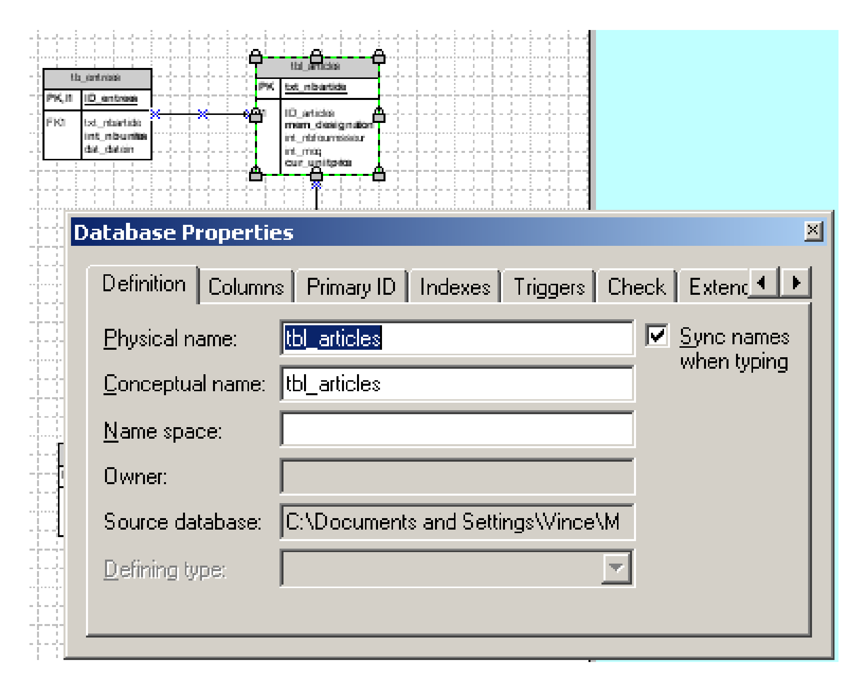
Figure 4 Modèle Logique de Données (MLD) dans MS Visio

Sur la gauche de l'écran, se trouvent les deux formes "Entity" et "Relations" nécessaires à la modélisation des données de la page suivante:

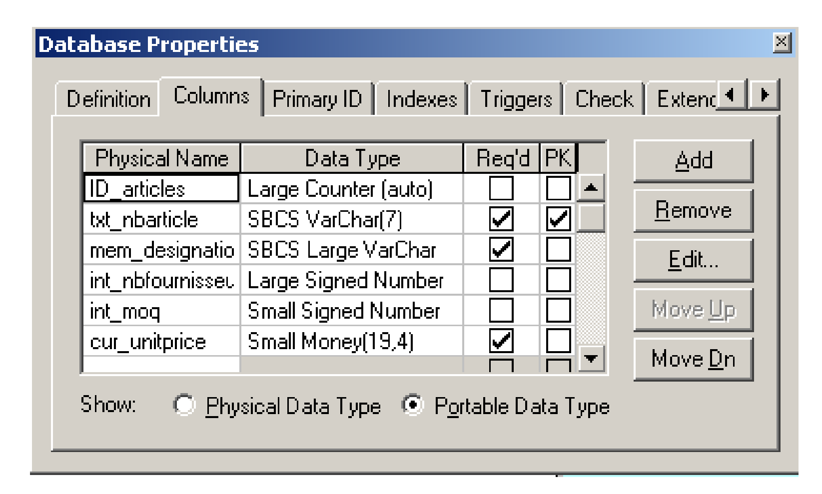


Légende: PK = Primary Key, FK = Foreign Key, I = Index

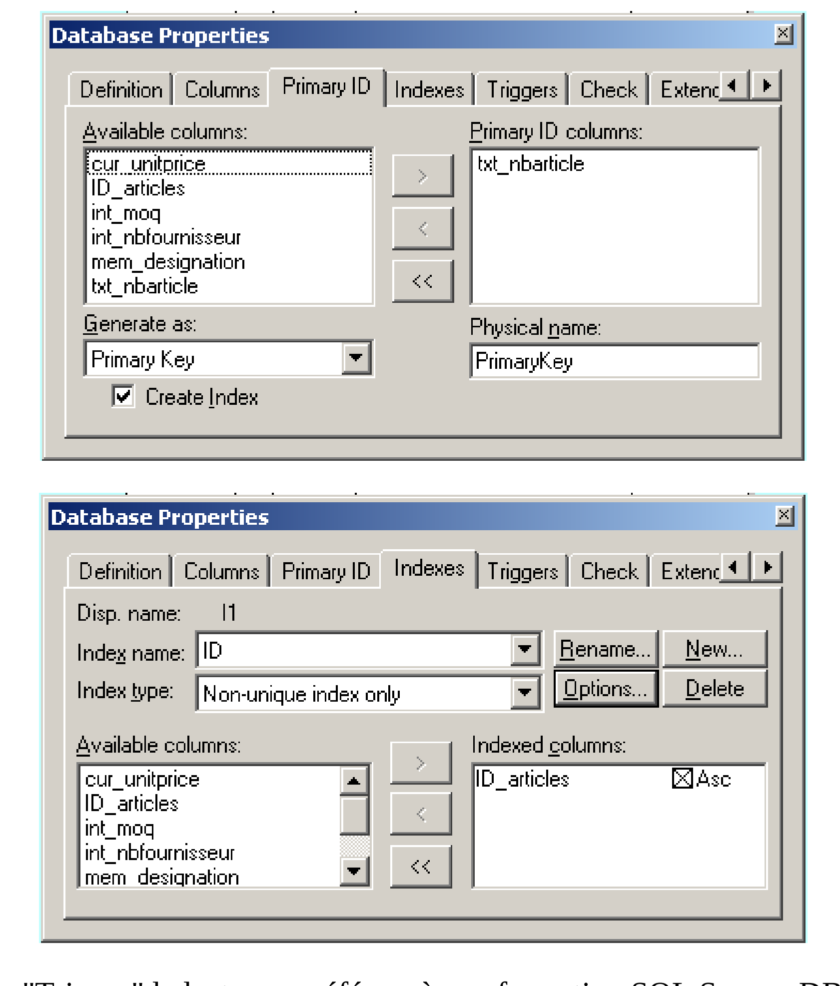
Lors de l'insertion d'une table, après un double clic sur cette dernière (nous avons pris pour exemple ci-dessous la table "tbl_sorties") apparaît la fenêtre pop-up suivante:



Et voici les propriétés des autres registres (voir page suivante):



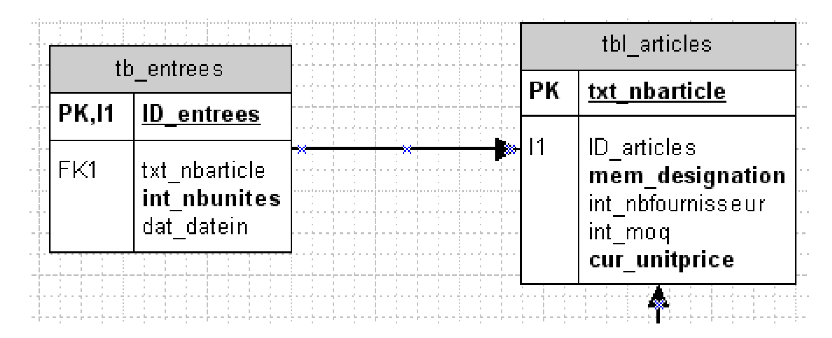
Concernant le typage des colonnes ci-dessus, celui-ci a nettement changé dans les ultérieures. Il a été standardisé pour proposer une compatibilité vers différentes normes SQL, raison pour laquelle on trouvera par exemple les typages *LONGTEXT* et *LONGCHAR* (ce dernier étant à privilégier pour MS Access) qui pour un forward engineering vers MS Access se retrouveront de toute façon convertis en type *Mémo*.



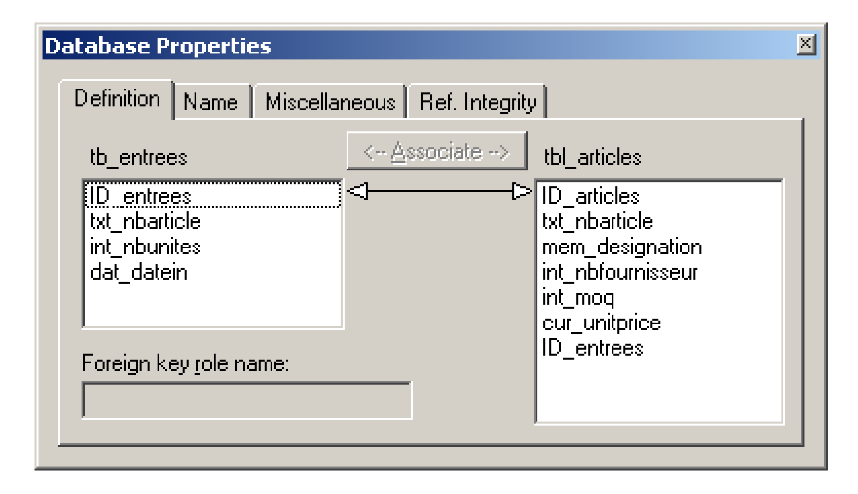
Pour le registre "Trigger" le lecteur se référera à une formation SQL Server, DBL2, Oracle. Pour le registre "Check" l'auteur de ce document donne sa langue au chat… Les deux autres

registres "Entended" et "Notes" ne sont pas absolument indispensables, nous en ferons donc pas mention.

Concernant les relations, il suffit de "glisser" une forme représentative de cette dernière entre les deux tables et de l'y accrocher (il faut que les tables deviennent alors rouge).

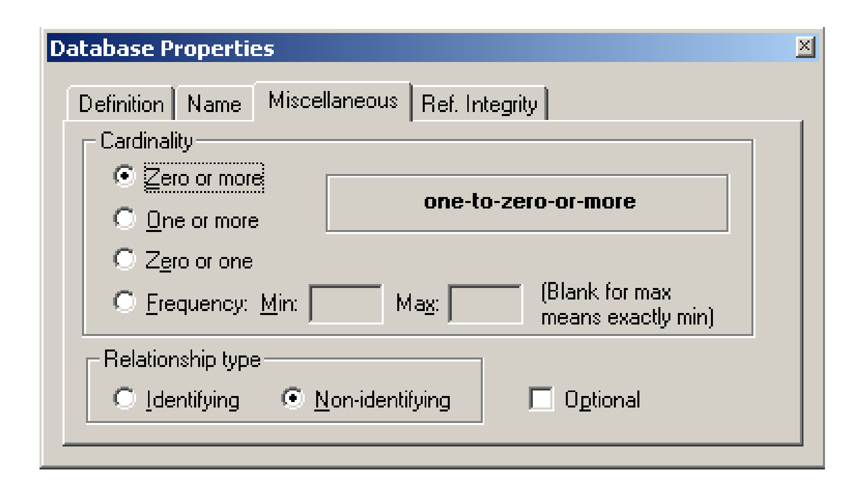


Pour relier la clé primaire (PK) *idArticles* de la table *tblArticles* à la clé étrangère *strNbArticles* (oui… la dénomination des champs est mauvaise…sur les captures d'écran) de *tblEntrees* (un article peut être entré plusieurs fois…) il suffit par un double clic sur la relation pour faire apparaître la boîte suivante:



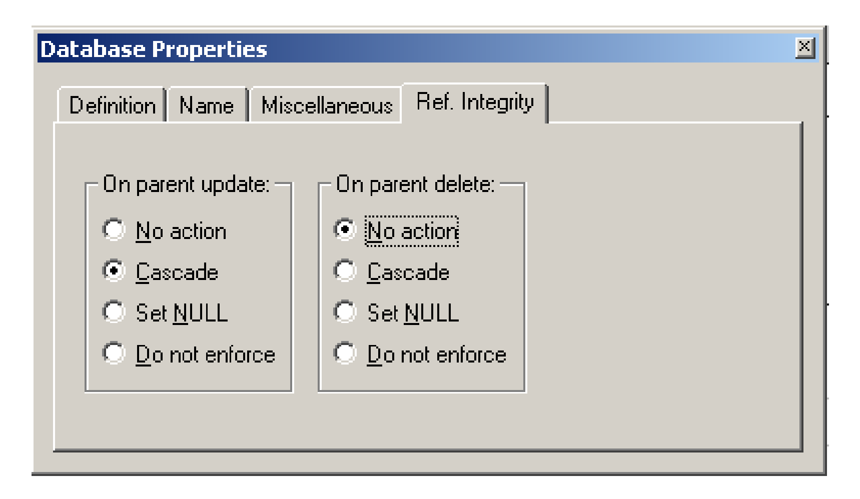
Avec la touche Ctrl du clavier, sélectionnez dans la liste de droite et de gauche, les champs à lier et cliquez sur le bouton *Associate*.

Ensuite, vous pouvez dans *Misceallaneous:*



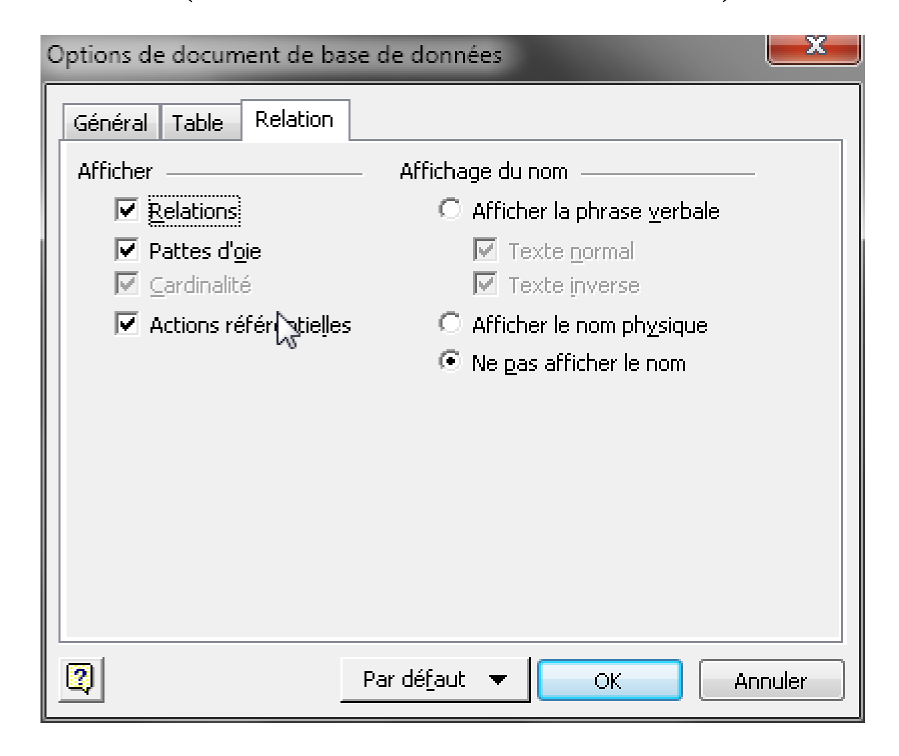
Nous voyons que dans MS Visio, les options proposées vont plus loin que celles proposées par MS Access seul.

En ce qui concerne l'intégrité référentielle:

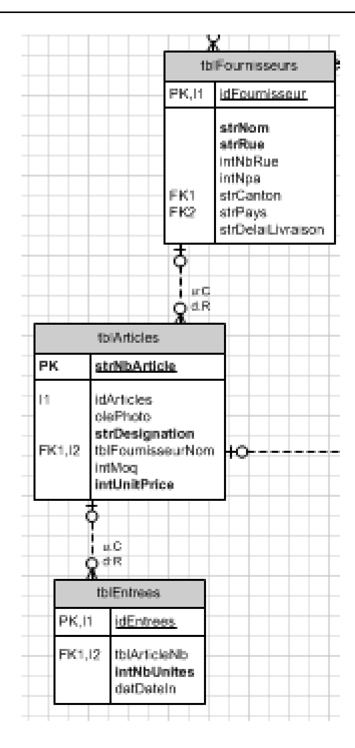


Pour exporter le schéma dans une base, il faut d'abord avoir créé le support logique de cette dernière. Dans MS Access cela consiste, par exemple, à créer un fichier *.mdb vide (exemple: *visio.mdb*).

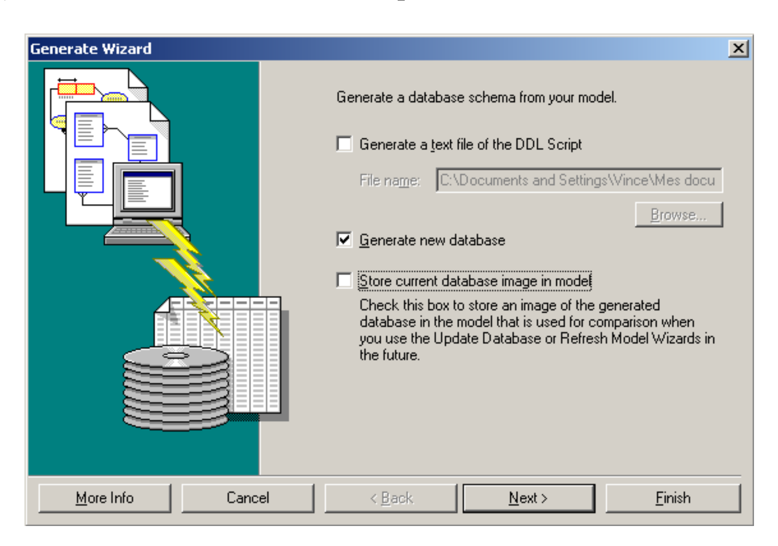
Enfin, nous remarquons qu'il est désagréable de ne pas avoir la cardinalité des relations, pour cela heureusement dans le menu *Database/Options* nous avons une boîte de dialogue permettant de les afficher (nous conseillons vivement de tout cocher):



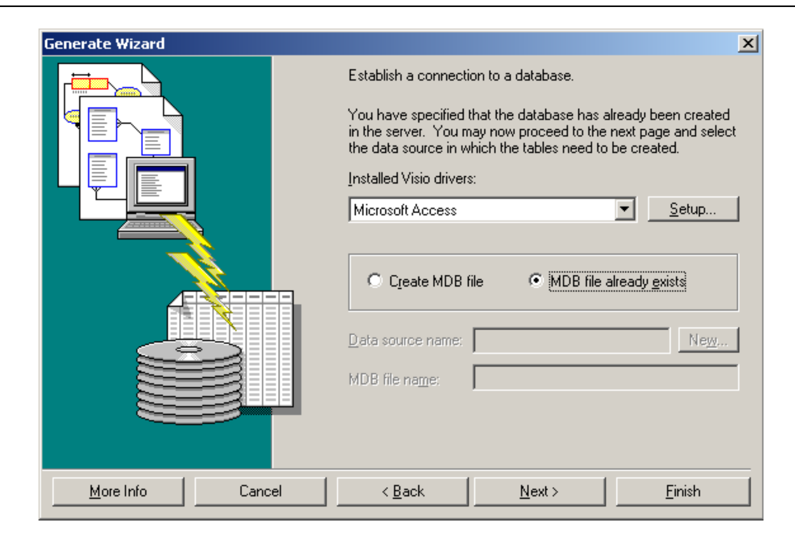
Dès lors nous avons:



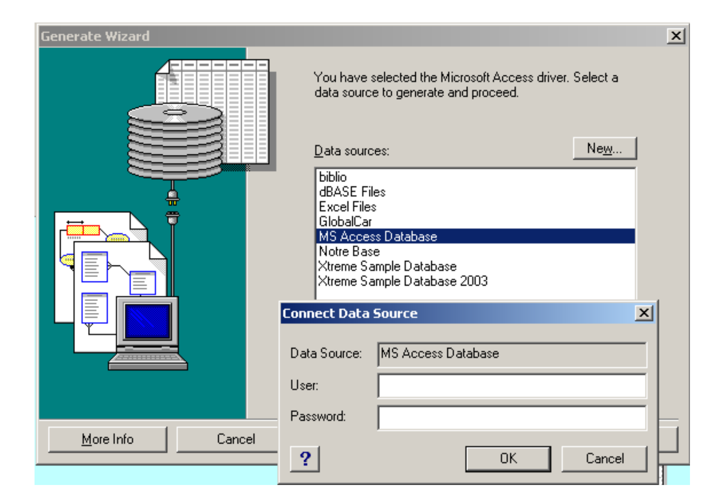
Avec MS Visio, allez dans le menu *Database/Generate* (avec MS Visio 2003 et antérieur puisque depuis 2007 il faut Visual Studio avec Visio Enterprise Architect...). Apparaît alors la fenêtre suivante (si aucune erreur n'est détectée par MS Visio dans votre modélisation):



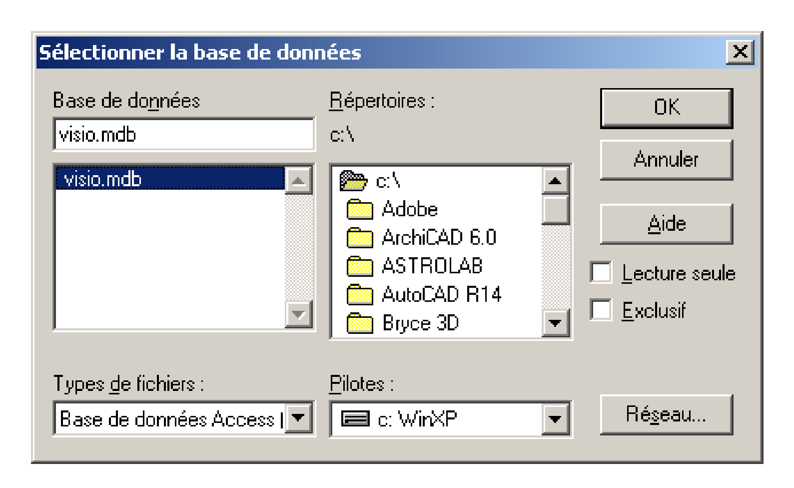
Cliquez sur *Next* après avoir activé les mêmes options que dans la figure ci-dessus.



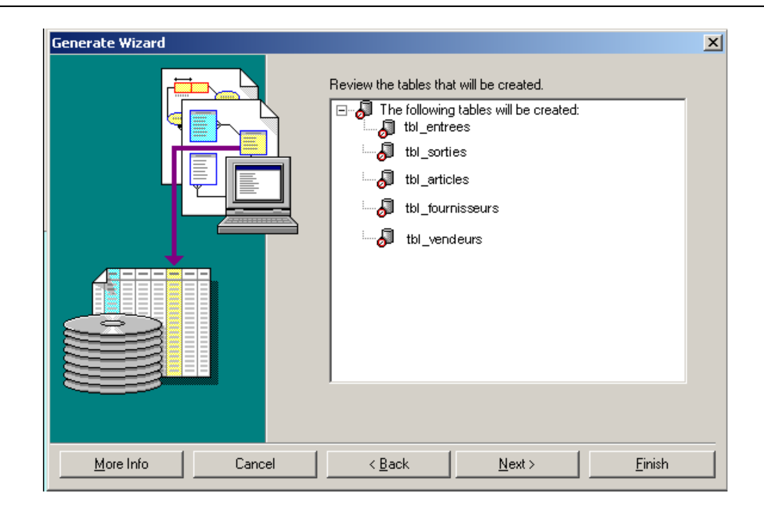
Sélectionnez *MDB file already exists* et cliquez sur *Next* autant de fois que nécessaire pour arriver à la fenêtre suivante:



Cliquez dans la boîte ci-dessus sur *OK* sans saisir ni *User* ni *Password*. Allez ensuite chercher la base cible:



Cliquez sur *OK:*



Cliquez enfin sur *Next* et *Finish*. Ouvrez le fichier *visio.mdb* et vous y trouverez toutes les tables et relations modélisées dans MS Visio (c'est pas beau ça non !!??).

Suite à cette démonstration, nous vous demandons de faire deux exercices (au cas où MS Visio n'est pas disponible, nous vous demandons d'utiliser du papier grand format) en groupe(s) et de présenter ensuite votre résultat et vos conclusions au formateur ainsi qu'aux autres groupes.

Remarques: les notations seront les suivantes sur vos feuilles:

Tableau 2 Légendes Modélisation BDD

I = index	1 = one	CU = Cascade Update	LR = Right Jointure
PK = Primary Key	∞ = many	CD = Cascade Deletion	LS = Simple Jointure
FK = Foreign Key	RI = Referential Integrity	LJ = Left Jointure	

Veuillez également respecter la nomenclature pour les objets et champs MS Access (voir page suivante).

3.8 Nomenclature de Leszynski/Reddick

Pour le développement (base de données et autres), il y a certaines règles et traditions qu'il vous faut respecter dans un premier temps pour votre confort, dans un deuxième temps pour être compatible avec vos collègues et enfin, dans un troisième temps, pour éviter les problèmes de possibles futures migrations ou les confusions entre nom de fonctions réservées et noms de champs (une erreur typique étant de nommer un champ **Name, Time** ou **Second** car il s'agit aussi de fonctions et ce sont des dernières qui prendront le dessus dans les rapports/états).

Remarques:

R1. Une partie de cette nomenclature est inspirée de la norme ISO 9660

R2. Il y a plusieurs manières d'écrire la nomenclature Leszynski/Reddick…

Des exemples: www.mvps.org/access/general/gen0012.htm

Les règles Lezsynski/Reddick pour le développement de base de données sont les suivantes (elles ont été également adoptées pour d'autres langages de programmation):

1. Majuscule au début de chaque mot d'une variable
2. Pas d'accents
3. Pas de caractères spéciaux
4. Pas d'espaces
5. Nom des champs en anglais
6. Eviter de dépasser les 15 caractères (3+12)
7. Ne jamais commencer avec des chiffres!

Il faut aussi débuter avec cette nomenclature les noms des objets par trois caractères en minuscule tel qu'indiqué ci-dessous:

- Nom des tables: **tbl**….
- Nom des requêtes: **qry**…
- Nom des vues: **vue**… (pour les SGBDR qui en ont…)
- Nom des états (rapports): **rpt**…
- Nom des formulaires: **frm**…
- Nom des index: **idx**…
- Nom des macros: **mcr**…
- Nom des modules: **mod**…

Et les noms des champs de table par des identificateurs tels que ci-dessous:

- Nom des champs clés primaire avec numéro automatique: **id**NomTable**PK** ou **pk**NomTable
- Nom des champs clés étrangères: principalement fkNomTableNomChamp mais aussi **tbl**NomTableNomChamp ou **tbl**NomTableNomChamp**FK** ou encore (mais déconseillé) **lst**NomTableNomChamp
- Nom des champs texte: **str**Nom
- Nom des champs numériques entiers: **int**Nom
- Nom des champs numériques décimaux: **sng**Nom
- Nom des champs de date: **dat**Nom
- Nom des champs booléens (Oui/Non): **bol**Nom…
- Nom des champs objets (Object Linked and Embedded): **ole**Nom

- Nom des champs avec liens hypertextes: **str**Nom
- Nom des champs mémo: **blb**Nom ou **mem**Nom ou **str**Nom sont acceptés

Et les noms des objets de formulaires par des identificateurs tels que ci-dessous:

- Listes déroulantes: **lst**NomListe
- Groupe d'options: **opt**GroupeOptions
- Cases à cocher: **chk**ChoixCase
- Boutons bascules: **tgl**ToggleButton
- Champs: **fld**NomChamp
- Bouttons: **btn**Button
- Onglets: **reg**Registres
- Légendes: **lgd**Legendes

Le respect de ces règles vous permet d'assurer une cohérence dans votre travail et une compatibilité sûre avec des systèmes autres que MS Access en cas de migration de la base.

Le non-respect de ces règles peut entraîner des conséquences telles qu'un énorme travail pourrait être à refaire dans la base de données !!!

4 Tables

Avant de commencer à travailler voici quelques informations:

1. Il existe plusieurs fichiers de base de données sous MS Access: *.mdb, *.adp, *.mde et les données peuvent provenir de plusieurs sources compatibles gérées par MS ADO (Microsoft ActiveX Data Objects). tel que MS Excel, DB2, Exchange, SQL Server, etc…

   Pendant l'ensemble de cette formation, nous allons travailler sur des fichiers *.mdb. Les fichiers *.adp sont destinés plutôt à des développeurs (informaticiens) qui utilisent SQL Server comme gestionnaire de bases de données relationnelles (SGBDR).

   Cependant, si vous créez un fichier *.adp (Access Database Project), la fenêtre (bien connue par les développeurs) suivante apparaît, qui vous demande à quelle base de données SQL Server vous désirez vous connecter:

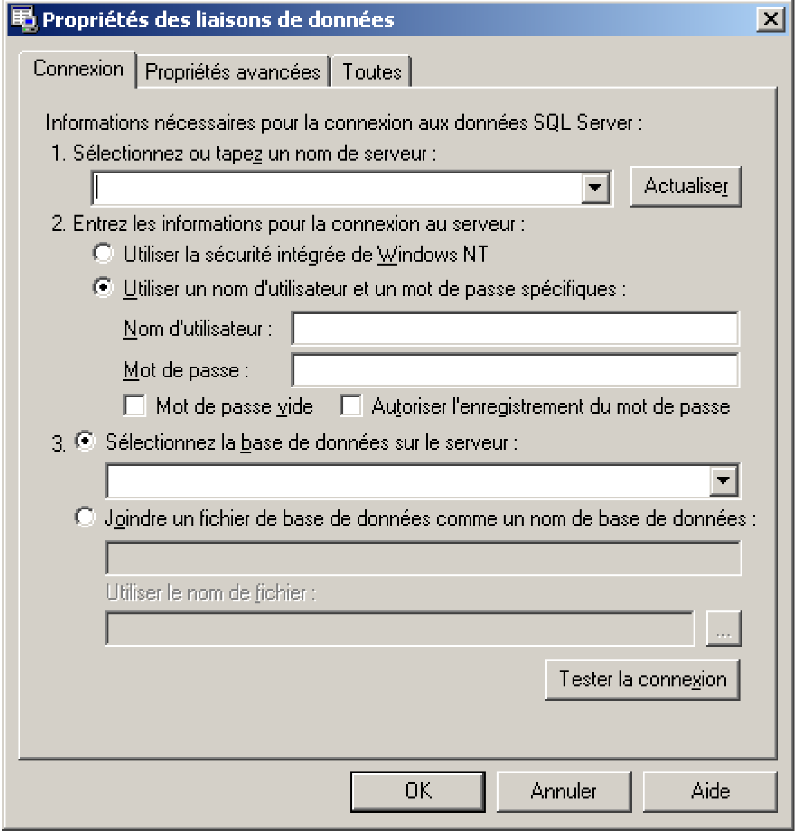


Figure 5 Accès BDD à distance (ADP)

Remarques:

R1. Il existe plusieurs technologies de SGBDR (FileMaker, SQL Server, MySQL, Oracle, AS400,…)

R2. MS Access est un environnement gérant ADO, SQL, VBA, XML, etc.

R3. Une fois une erreur effectuée ou des enregistrements effacés, on ne peut plus revenir en arrière!!!

R3. La provenance des tables de données, dans MS Access, peut être diverses:

1. Cas simples: base MS Excel Externe, base MS Access Externe
2. Cas plus complexes: Oracle, MySQL, DB2, SAP, ou autres…

A chaque fois que vous créez un objet en utilisant un des logiciels de la suite MS Office il faudrait définir les propriétés de ce dernier. Microsoft Project et Visio le font automatiquement. Ce n'est malheureusement pas le cas de MS Access. C'est pourquoi nous allons tout de suite nous attarder à cette tâche. Dans le menu *Fichier*, sélectionnez l'option des Propriétés et saisissez les informations suivantes:

Propriétés de Bibliothèque.mdb

Général | Résumé | Statistiques | Contenu | Personnalisation

Titre : Magasin
Sujet : Gestion du Magasin
Auteur : Votre Nom Votre Prénom
Responsable : Le nom du responsable de Projet
Société : Votre société
Catégorie : Gestion de stocks
Mots clés : entrées, sorties, analyses, comptabilité
Commentaires : Cette base évolue constamment et sa construction peut être répartie sur 10 jours de formation.
Répertoire Web :
Modèle :

OK | Annuler

La fenêtre d'explorateur d'objets sera votre outil principal pendant votre travail dans Access.

Vous pouvez depuis cette fenêtre:

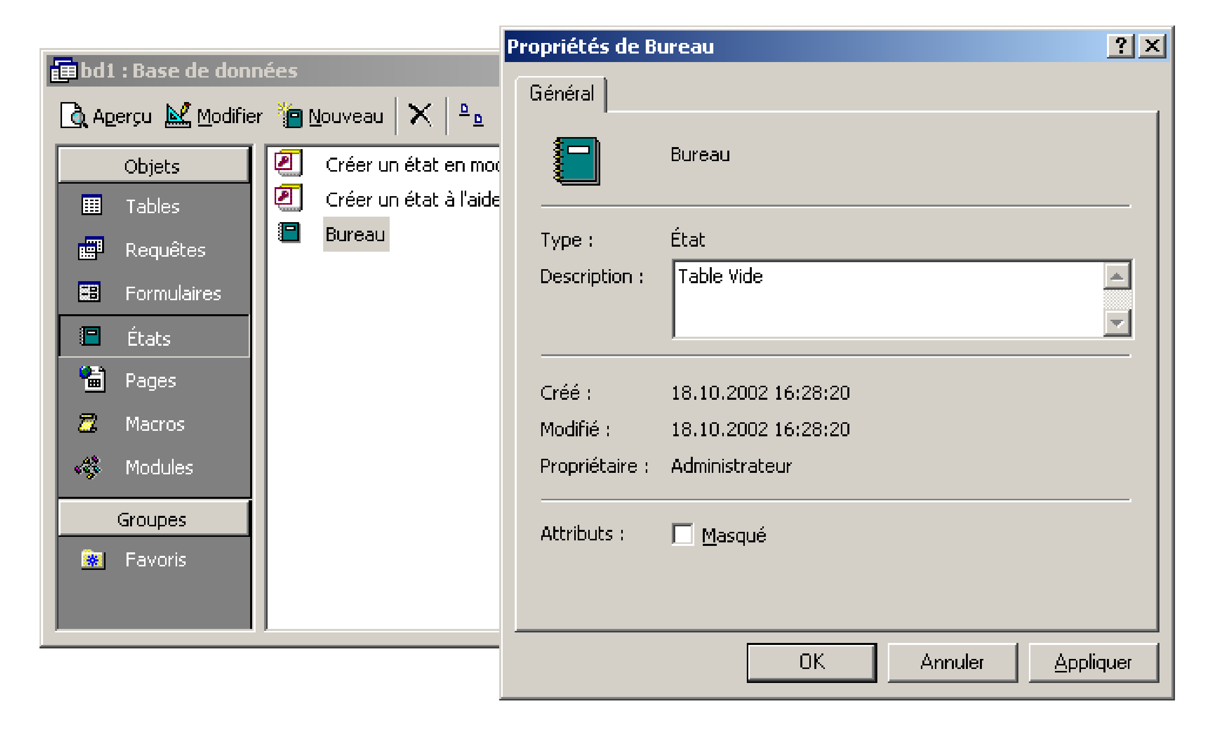
Lancer l'aperçu, modifier, créer, supprimer, copier, renommer, exécuter, masquer et afficher des objets.

Avant que nous attaquions la création d'une base, il est nécessaire de définir les objets disponibles dans Access (formulaires, tables, requêtes, états, macros, modules et pages)

Il arrive lorsque l'on travaille sur d'énormes bases, que l'explorateur d'objets soit remplis d'un trop grand nombre d'objets pour les visualiser d'un simple regard. On choisira alors parfois de masquer les objets (quels qu'ils soient) dont on est certain que l'on ne modifiera probablement jamais les propriétés et le contenu.

On peut masquer un objet en cliquant dessus avec le bouton droit de la souris et en sélectionnant l'option "Propriétés" et en cochant la case "Masquer".

Pour afficher à nouveau les éléments masqués, il suffit d'aller dans *Outils/Options* l'onglet *Affichage* et activer la case *Objets cachés*.



Les tables ont-elles quelques cases à cocher en plus comme *Réplicable* et *Suivi du nombre de lignes* mais ce n'est que pour les utilisateurs travaillant avec les options de réplication.

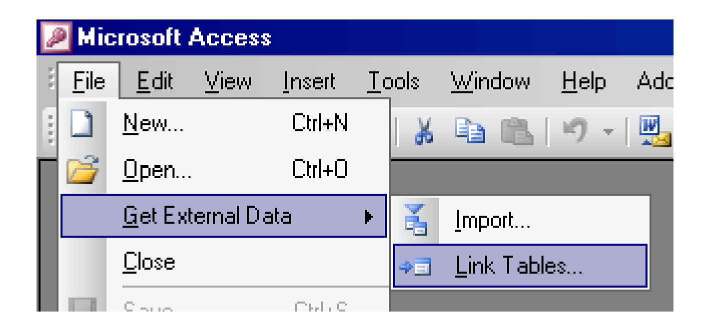
Si vous souhaitez utiliser cette option pour empêcher certaines personnes de modifier votre base, ne procédez pas ainsi. Nous verrons plus tard comment il est possible de protéger sa base avec un mot de passe afin que personne d'autre que vous ne puisse la modifier.

4.1 Liaison MS Excel/CSV

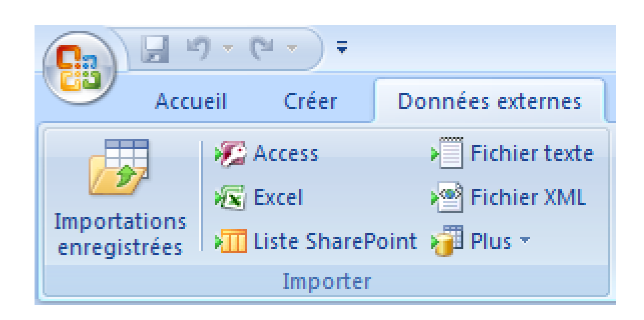
Nous allons tout de suite voir comment lier une feuille MS Excel de deux manières différentes (cette table sera utilisée bien plus loin dans le cadre du cours intermédiaire/avancé) avant même de créer une table par la méthode standard:

1. Importer physiquement et définitivement une table MS Excel:

Dans MS Access allez dans *Fichier/Source de données externe/Importer* (*File/Get External Data/Import*):



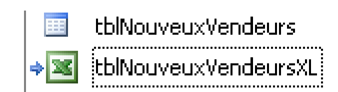
et sélectionnez dans *Type de fichier* le type *MS Excel*. Sélectionnez le fichier *TableNouveauxVendeurs* et suivez simplement l'assistant. Observez l'icône qui représente la table… dans MS Access 2007 et ultérieur dans le ruban *Données Externes/Fichier Texte*:



et suivez l'assistant…

2. Lier une table MS Excel de manière dynamique:

Dans MS Access allez dans *Fichier/Source de données externe/Lier* et sélectionnez dans *Type de fichier* le type MS Excel (remarquez que vous pouvez aussi vous connecter à MS Outlook comme une base de données !!!) Sélectionnez le fichier TableNouveauxVendeurs.xls et suivez simplement l'assistant. Observez l'icône qui représente la table.



Modifiez le fichier MS Excel et ouvrez ensuite la table liée dans MS Access pour voir si la liaison s'effectue bien (vous pouvez ajouter/supprimer/modifier les lignes ou colonnes dans le fichier source à votre guise!). Vous verrez que MS Access ne peut pas modifier les propriétés des champs de cette table liée (donc pas de clé primaire, pas de relations, etc…)

Attention, sur ce genre de tables liées, vous ne pouvez:

1. Pas créer de clés primaires et donc de liaisons (mais vous pouvez quand même créer des assistants liste de choix)

2. Vous ne pouvez pas appliquer de requête de suppression

3. Vous ne pouvez plus renommer les champs une fois l'import effectué

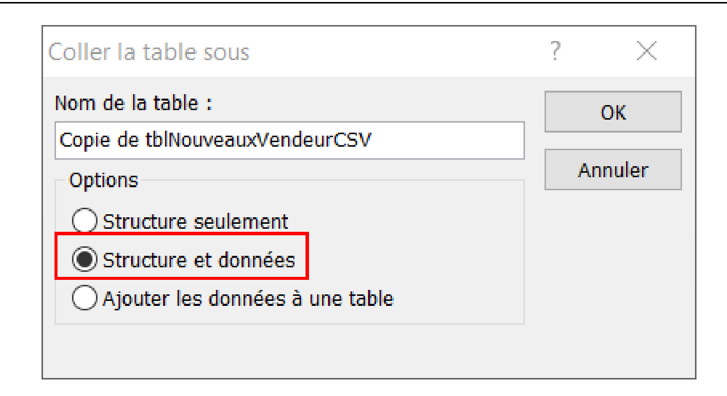
4. Vous ne pouvez pas ajouter ou modifier des champs (colonnes)

5. et autres que vous découvrirez par vous-même…

L'import d'un fichier CSV a une icône particulière que voici:



Remarque: La manière la plus rapide et simple de faire d'une table connectée, une table locale, c'est de copier/coller la première et lors du collage choisir l'option suivante:



4.2 Création de tables

Nous allons travailler sur une base qui permet de gérer le stock des produits d'un magasin.

Créer une nouvelle base nommée: Magasin.mdb (fichier *mdb* pour "Microsoft Data Base"). Dans le cas où vous êtes sur une version antérieure d'Access faites apparaître les barres d'outils de Formatage et Feuille de données.

1. Créez une nouvelle table en mode *Feuille de données*
2. Entrez tous les noms et données de champs ci-dessous dans les colonnes 1 à 5
3. Ajustez correctement la largeur des colonnes
4. Modifiez le type de données des champs (Texte, Nombre, Monétaire)
5. Saisissez une description pour chaque champ et sa Légende dans les propriétés du champ

N° Article	Désignation	N° Fournisseur	Quantité par unité de commande	Prix unitaire
GEN-001	Crayons	1	100	SFr. 0.20
GEN-002	Enveloppes (10 pces)	2	50	SFr. 0.50
GEN-003	DIN A4 Papier (500 feuilles)	3	50	SFr. 23.85
GEN-004	Post-It Notes 656	1	60	SFr. 9.80
GEN-005	Post-It Notes 657	1	60	SFr. 10.40
INF-001	Tambour	3	5	SFr. 249.00
INF-002	D2isquette (3.5")	3	1000	SFr. 1.30
INF-003	Etiquettes Laser (25 feuilles)	3	10	SFr. 35.90

On mettra les détails des fournisseurs dans une autre table, où ils seront reconnus pas des numéros.

Pour créer la clé primaire de la table:

1. Faire passer la table en "mode création"
2. Sélectionner la ligne voulue comme clé primaire (*strArticleNb*)
3. Cliquer sur l'icône de la barre d'outils représentant une clé
4. Enregistrer la table sous le nom *tblArticles*

La puissance d'un programme de gestion de bases de données relationnelles comme MS Access provient de sa capacité à trouver et réunir rapidement des informations stockées dans des tables séparées en utilisant des requêtes, des formulaires et des états. A cette fin, chaque table doit inclure un champ ou un ensemble de champs qui identifie, de manière unique, chaque enregistrement stocké dans la table. Cette information est appelée la "clé primaire" de la table. Une fois que vous avez désigné une clé primaire pour une table, Microsoft Access empêchera, pour en garantir le caractère unique, que des doublons ou des valeurs "Null" ne soient entrés dans les champs de clé primaire.

Si un de vos champs contient des valeurs uniques, comme des numéros d'identification ou des numéros de pièces, vous pouvez désigner ce champ comme clé primaire. Cependant, si ce champ contient des doublons ou des valeurs Null, Microsoft Access ne l'acceptera pas comme clé primaire.

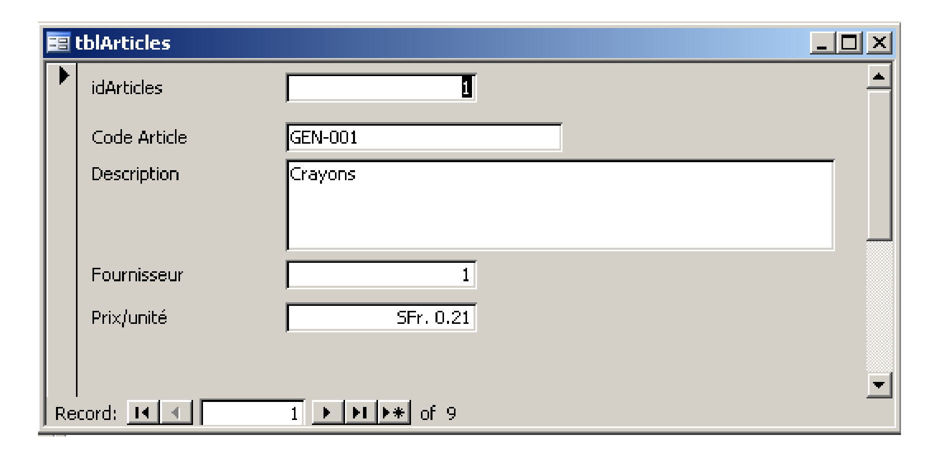
Vous pouvez alors exécuter une requête "Trouver les doublons" pour trouver les enregistrements contenant des doublons (nous verrons cela lors de notre étude des requêtes). Si vous ne parvenez pas à éliminer facilement les doublons en éditant vos données, vous pouvez ajouter un champ nommé "numéroter automatiquement" (insérer une ligne en mode création et définir le type de données comme "Numérotation automatique") et le définir comme clé primaire, soit définir une clé primaire combinée.

4.3 Formulaire simple (auto-form)

Pour créer votre premier formulaire:

Aller dans la fenêtre de *Base de données* et cliquer sur le bouton *Formulaires* et choisir *Créer un formulaire à l'aide de l'assistan*t. Une fois un formulaire créé, compris et personnalisé un peu avec l'aide du formateur (n'oubliez pas de définir la propriété "Légende"), fermez-le sans l'enregistrer.

Créez un formulaire instantané pour la table *tblArticles* (dans le menu Fenêtres il y a une commande fort utile pour les vieux ordinateurs ayant une résolution d'écran très inférieure: Ajuster à la taille du formulaire. Il faut évidemment au préalable avoir ouvert le formulaire sur lequel on veut activer cette option):

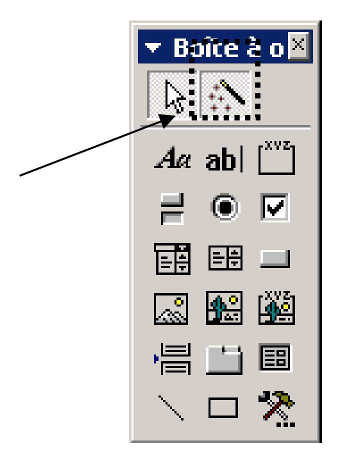


Vérifiez les enregistrements existants, et ajoutez les enregistrements suivants:

GEN-006	Stylos rouges	1	100	SFr. 0.50
INF-004	Toner	3	20	SFr. 85.90

Remarque: L'enregistrement n'est sauvegardé qu'au moment où le point d'insertion est déplacé sr un autre enregistrement (sinon il y a possibilité d'ajouter un bouton pour cela mais c'est trop tôt pour en parler de manière détaillée).

Supprimez maintenant l'enregistrement n°5 (il y a 3 possibilités pour cela: 1. Menu 2. Bouton standard 3. Création de bouton) et réinsérez-le à l'identique. Lorsque vous créez un bouton, faites bien attention à ce que l'assistant de la boîte à outils contrôles soit bien activé !



Concernant les ordres de tabulation des champs de données du formulaire: vous pouvez les définir (peut s'avérer parfois très utile) en allant en mode création (quand votre formulaire est ouvert) et en sélectionnant ensuite le menu Affichage/Ordre de tabulation[5].

Fermez le formulaire. Enregistrez le sous le nom *frmArticles.*

Créez une table *tblFournisseurs* avec les données de la table ci-dessous. Les numéros de fournisseurs serviront de clé primaire numérotée automatiquement

Fournisseurs : Table

N° Fournisseur	Nom du fournisseur	Rue	NPA	Lieu	Délai de livraison/Jours
1	Duplibureau	Ruelle du soleil	2003	Neuchâtel	5
2	La maison du papier	Rue des Anges 17	1010	Lausanne	8
3	Tartanpion Fournitures	Grand-Rue 115	1205	Genève	3

Nous "relierons" plus tard la table *tblFournisseurs* et *tblArticles* afin que lorsqu'on insère de nouveaux articles à l'aide du formulaire, on ait le choix des numéros des fournisseurs (on verra dans le cours expert comment faire en sorte que le nom du fournisseur apparaisse automatiquement dans un champ lorsque l'on sélectionne son numéro).

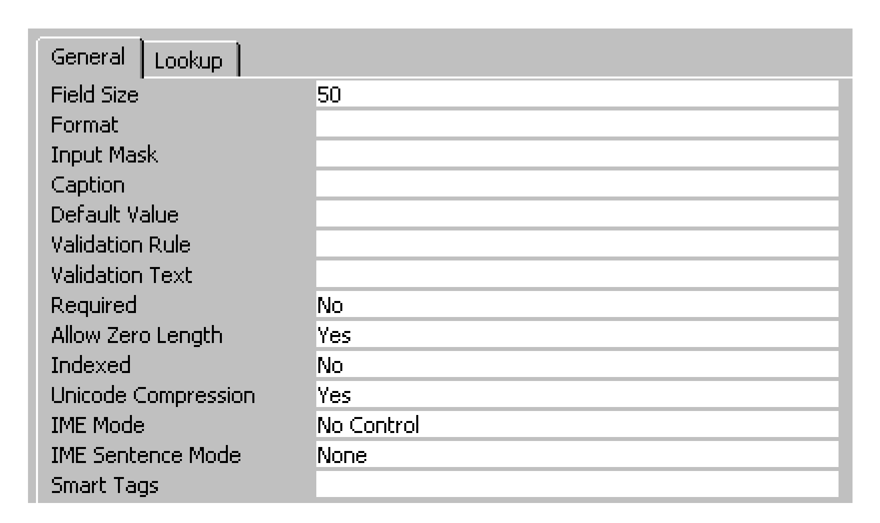
4.4 Contrôle des données

Comme nous l'avons vu, lorsque vous passez une table en mode création Design, il y a plusieurs zones de champs dans lesquelles l'on peut saisir du texte ou des options dans des menus déroulants. Voyons quelques-unes des possibilités qui nous sont offertes (il y aura des exercices là-dessus, de suite après la théorie):

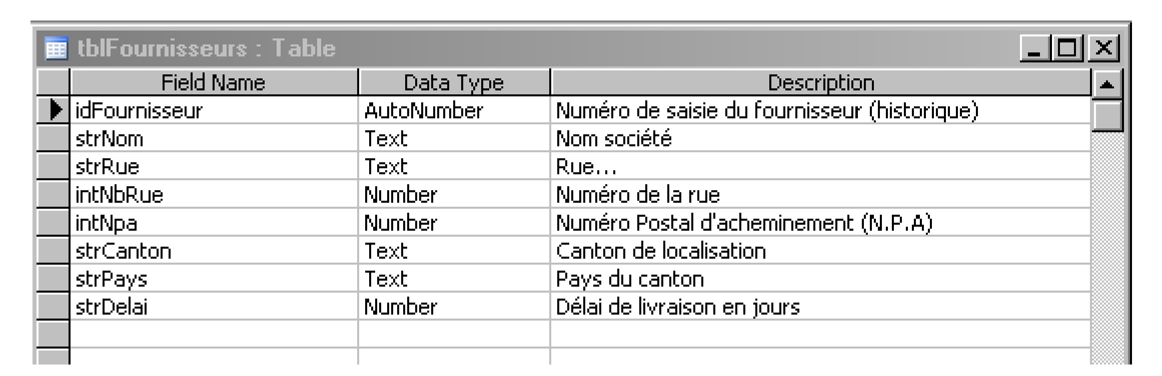
[5] Notez que les boutons (mais pas que!) ont une propriété qui se nomme Arrêt tabulation afin d'éviter qu'ils fassent partie de l'ensemble des éléments sur lesquels la touche tabulation a un effet.

4.4.1 Légende

Donc dans le mode création vous aurez pour chaque champ la zone suivante:

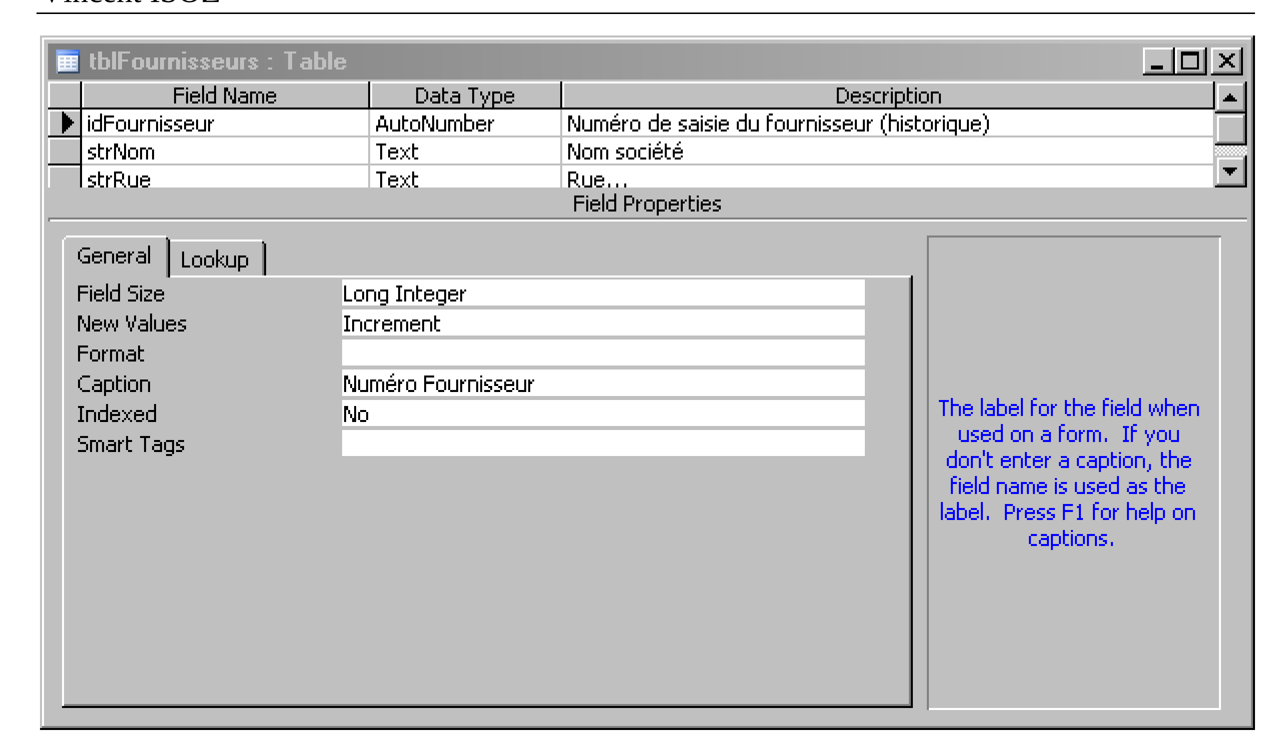


Ce qui nous intéresse ici en priorité (pour faire simple) c'est le fait que nous devons respecter la nomenclature de Leszinsky/Reddick. Il est donc important pour chaque champ créé précédemment de changer les noms des champs de manière conforme:


tblFournisseurs : Table

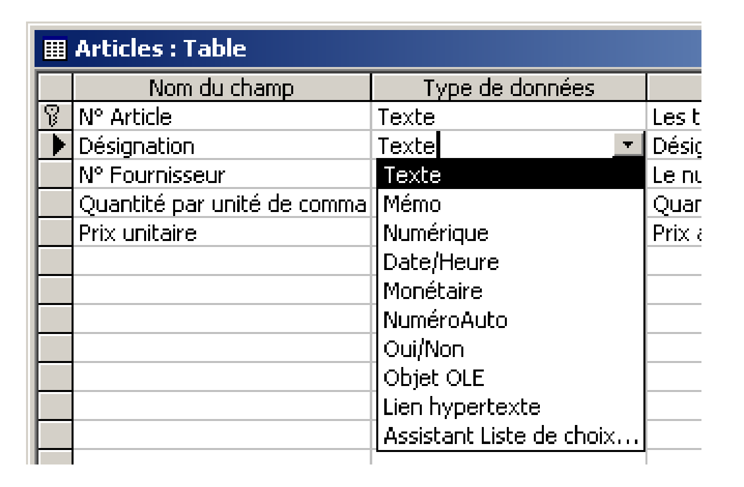
Field Name	Data Type	Description
idFournisseur	AutoNumber	Numéro de saisie du fournisseur (historique)
strNom	Text	Nom société
strRue	Text	Rue…
intNbRue	Number	Numéro de la rue
intNpa	Number	Numéro Postal d'acheminement (N.P.A)
strCanton	Text	Canton de localisation
strPays	Text	Pays du canton
strDelai	Number	Délai de livraison en jours

et ensuite pour chacun de mettre un *Caption* (en français une *Légende*) qui correspondra au nom que l'utilisateur verra dans lors de l'utilisation de la base de données:



4.4.2 Types de données (Typages)

Les types de données sont accessibles dans la liste déroulante de la colonne *Type de données*:



Texte (Court) - Text

0 à 255 caractères alphanumériques au format brut (plain text) codé sur 8 bits et utilisent *str*… comme préfixe d'usage.

Il est important d'optimiser au mieux la taille des champs lorsque vous travaillez sur des base de données de plusieurs millions de lignes!

Mémo - Memo

Texte long sur 65'536 caractères alphanumériques que l'on peut formater que de manière élémentaire dans les versions antérieures à 2007 et de manière plus fine depuis MS Access 2007 (couleurs sur une partie du texte seul, idem pour le gras, idem pour l'italique, gestion du surlignage, etc.) si l'on met une des propriétés du champ au format *RTF*.

Le champ memo utilise *mem…* ou parfois *blb…* comme préfixe.

Numérique - Number

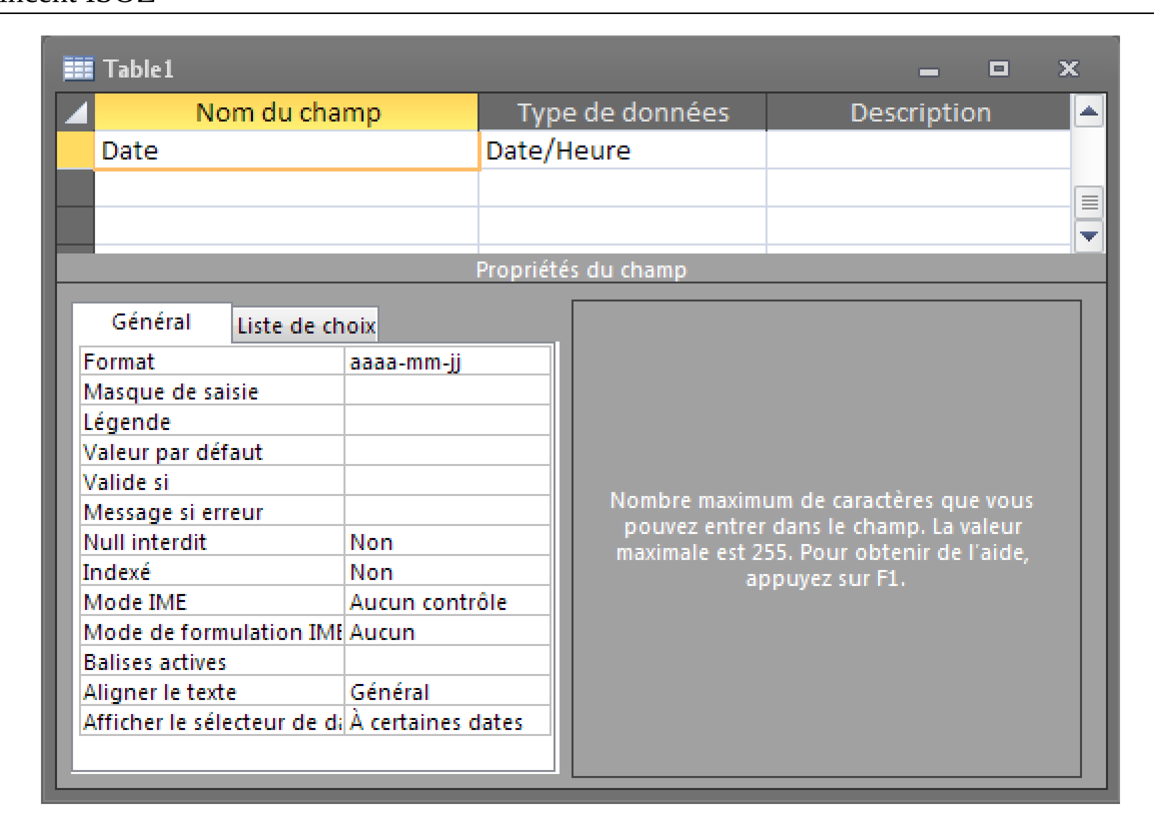
Type	*Caractéristiques*
Byte (Octet) | byt…:	0 à 255 (1 byte)
Integer (Entier) | int…:	-32'768 à 32'767 (2 bytes)
Long Integer (Entier Long) | int…:	-2'147'483'648 à 2'147'483'647 (4 bytes)
Single (Réel Simple) | sng… :	$-3.4.10^{38}$ à $3.4.10^{37}$ (4 bytes)
Double (Réel Double) | dbl…:	$-1.797 \cdot 10^{308}$ à $1.797 \cdot 10^{308}$ (8 bytes)
Decimal (Décimal) | dec…:	Nombre défini dans MS Access par une "précision" (nombre de chiffres au total qui peuvent apparaître dans le nombre avant **et** après la virugle) allant de 1 à 28 et par une "échelle" correspondant aux nombres de chiffre qui apparaîtront après la virgule en tant que décimales. Par exemple un décimal de 15 avec précision de 8, pourra avoir 7 chiffres avant la virgule et 8 après la virgule. Il n'est pas conseillé de faire usage du type Décimal car lors de l'utilisation de grands nombres les arrondis peuvent avoir des conséquences très fâcheuses!!

Date/Heure - Date/Time

Une date ou une heure (une précision à la seconde seulement…). Il existe plusieurs formats de saisie à choix dans les options générales de ce type de champ. A remarquer que ce dernier se comporte comme MS Office Excel si la saisie est au format JJ.MM.AAAA on peut quand même écrire *20 mai 2001* qui se changera alors automatiquement en *20.05.2001*.

Les dates vont du 01.01.100 au 31.12.9999 de 00:00:00 à 23:59:59 codé sur 8 bytes et utilisant *dat…* comme préfixe d'usage.

Attention!!! Rappelons que depuis 1988 tout les pays du monde utilisant le calendrier Grégorian doivent respecter la norme ISO 8601. Donc nous recommandonc donc de formater les champs en conformité:



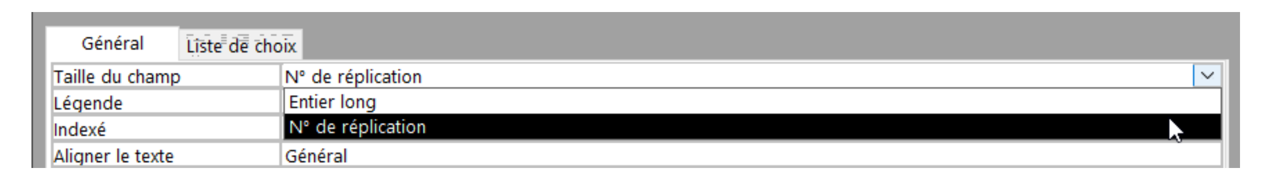
et même d'aller plus loin en s'assurant que les utilisateurs ne confondent jamais les mois et les jours à la saisie (par exemple 2003-11-03 sera accepté ainsi que 2003-03-11 donc c'est très dangereux suivant le domaine d'activité) et alors de séparer la saise des dates dans **3 champs différents** avec les contraintes en adéquation (quitte donc à les réassembler plus tard dans les requêtes)!!!!

Monétaire - Currency

Il s'agit au fait simplement d'un numérique avec un formatage monétaire. Codé sur 8 bytes avec 4 chiffres après la virgule. Je ne recommande pas d'en faire usage pour des raisons d'intercompatibilité avec d'autres systèmes et je ne vais donc pas m'étendre plus sur le sujet (de toute façon il est basé sur un réel simple).

Numéroauto - Autonumber

Il s'agit d'un champ souvent utilisé pour les clés primaires et basé simplement sur un entier long (4 bytes) ou sur un GUID[6] appelé aussi N° de réplication:



son préfixe d'usage est *pk*…

[6] Aussi loin que nous le savons il semble que le GUID ne contiennet pas de codes correcteurs d'erreurs!

Dans les multinationales il est conseillé d'utiliser le GUID qui ressemble typiquement à:



tbl_Pays

pk_Pays	str_Pays
{1B630F7D-CB51-4DBB-93BF-C6A117B8A687}	France
{47851025-5127-416E-B43B-1524FA1F1076}	Espagne
{57FB2F97-B688-4A88-A54C-BAC27E6B0E6F}	France
{5BBD990E-41B4-4F37-B063-8B392261D070}	Portugal
{6AEADAA6-51D7-4B5E-AE42-AD832C2068DD}	Allemagne
{7D4B056A-077C-4EA7-B4D2-16062D0571FC}	Suisse
{933F1FE5-B6A4-4EA0-BAC6-5F3897A2843F}	Italie

Pour éviter des collisions dans de la fusion avec d'autres bases de données et préparer la base de données à une potentielle utilisaton en mode "réplica".

Attention!!! L'utilisation du GUID rend les *DLookUp* inopérants (et pas que!) et il faut donc très vite utiliser des VBA (langage de programmation de MS Access) pour obtenir des résultats qui en temps normal sont pourtant simples (c'est-à-dire avec un clé de type *Entier Long*).

Oui/Non - Yes/No

Il s'agit d'un champ booléen. Il peut contenir qu'une valeur binaire correspondant à Vrai/Faux ou 1/0. Il est codé que sur 1 bit et dans MS Access le Oui vaut -1 et le Non vaut 0… Son préfixe d'usage est *bol*…

Objet OLE - OLE Object

L'object OLE (Object Linked And Embedded) est un type de champ qui vous permet d'insérer ou de lier des fichiers externes dans la base. Cela peut être utile pour les entreprises ayant à faire un suivi des clients par rapports aux documents qu'ils ont tapés pour eux (bureaux d'avocats typiquement).

Depuis MS Access 2007 il est conseillé pour les documents d'utiliser plutôt le champ de type *Pièces jointes* et qui est moins instable que le champ **OLE**.

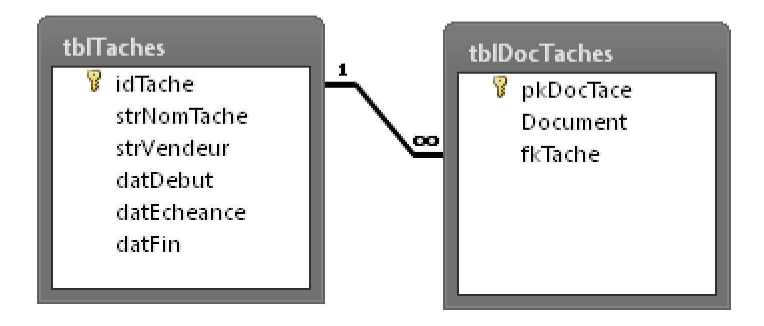
Son préfixe d'usage est *ole*…

Lien hypertexte - Hyperlink

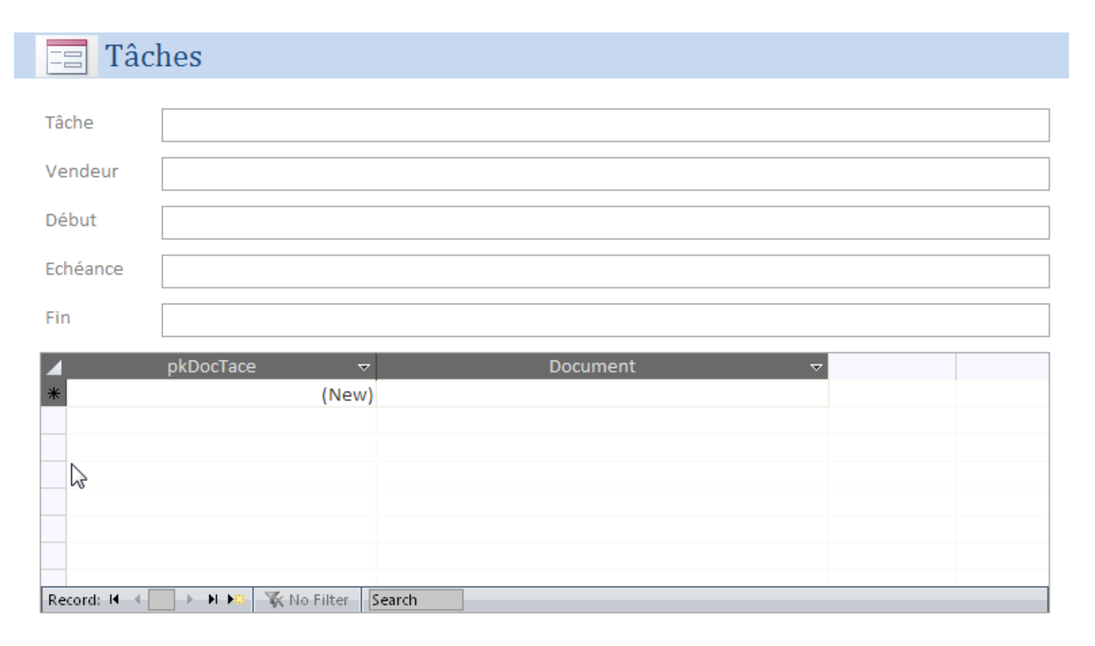
C'est un simple champ texte acceptant jusqu'à 2048 caractères qui a la propriété de fonctionner comme un lien hypertexte. C'est-à-dire que si l'utilisateur clique sur l'URL il sera envoyé sur la page web spécifiée. Le lien hypertexte ne fonctionne pas pour les e-mail comme c'est le cas dans MS Office Word et Excel et pose des problèmes lors de migration de bases MS Access vers d'autres technologies.

Le champ de type lien hypertexte peut cependant être très utile en tant qu'alternative au champ de type *Pièce Jointe* (voir un peu plus bas) qui est gourmand en matière de mémoire (puisqu'une base MS Access est limitée à 2GB).

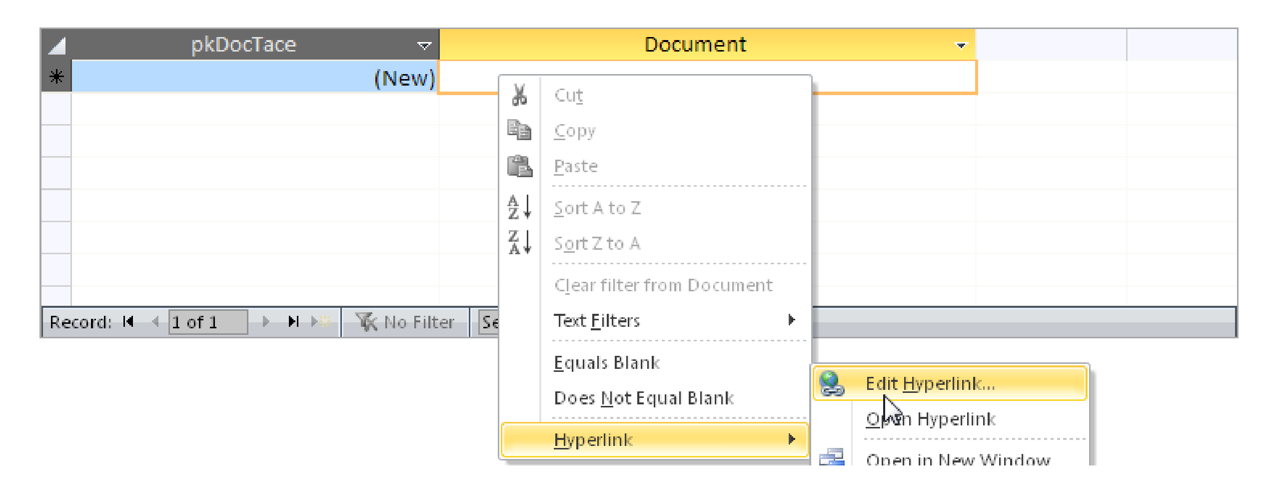
Une astuce possible est alors de créer une ou plusieurs tables contenant juste un champ de clé primaire et un champ de lien hypertexte en relation de type un à plusieurs et de créer des liens vers des documents se trouvant sur un disque réseau. Ce qui donnerait par exemple:



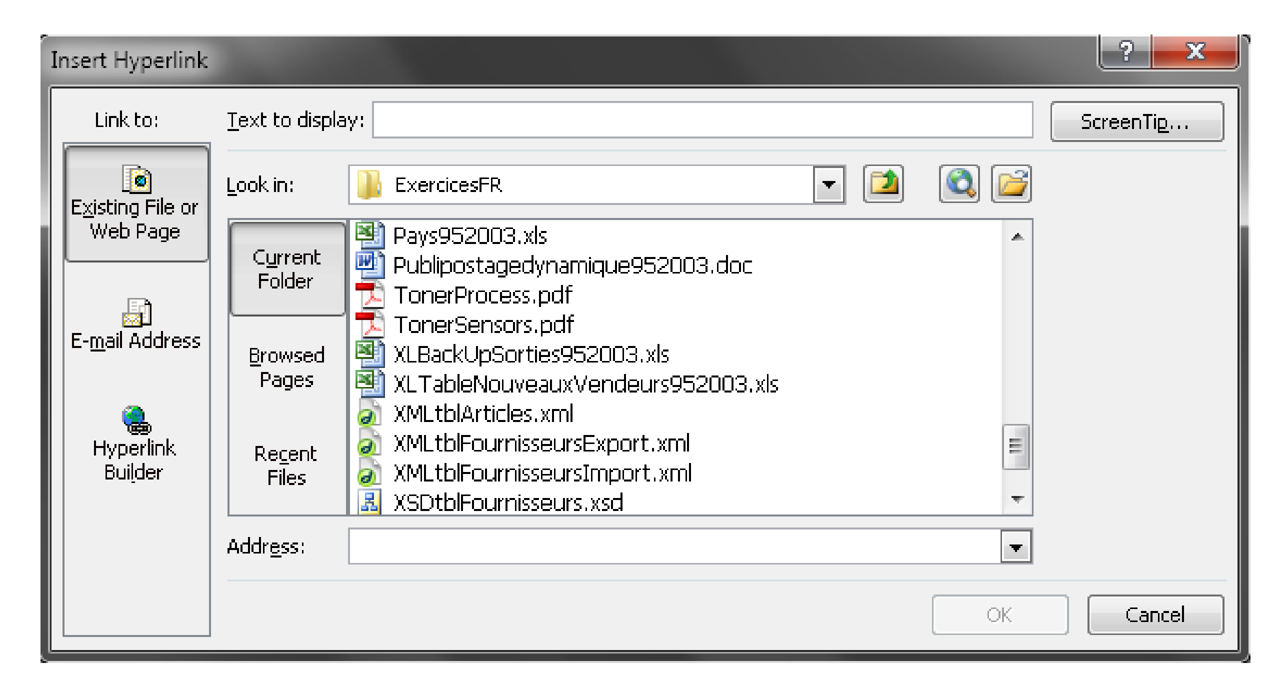
Ensuite, l'utilisateur aura typiquement un formulaire (vite fait mal fait...) du type suivant:



et pour créer un lien vers un document se trouvant sur un disque réseau il suffit de fair un clic droit une des cellules de la colonne *Document*:



et de sélectionner *Edit Hyperlink*... La suite est la même que dans MS Word/Excel/Outlook ou PowerPoint:



et donc il n'y a rien à en dire!

Son préfixe d'usage est *hyp*…

Assistant liste de choix - Lookup Wizard

Il s'agit d'un type de données dans lequel les valeurs possibles du champ peuvent être choisies d'après une liste (table) disponible et deviendra donc une clé étrangère basée majoritairement sur un entier log et écrit avec le suffixe fk…. La liste peut cependant aussi être définie individuellement.

Depuis MS Access 2007 il est possible de faire des champs multivalués (très utile pour les utilisateurs SharePoint). Bien qu'ils simplifient le travail de modélisation il faut si possible les éviter car le temps d'exécution des requêtes sur une grande quantité de données est parfois 10x plus long. + d'autres problèmes qui seront expliqués pendant la formation.

Pièces Jointes - Attachment

Type de champ non visible sur la capture d'écran car il s'agit d'une nouveauté de MS Access 2007. Ce nouveau typage gère l'insertion de pièces jointes multiples dans un document de manière stable (contrairement à OLE). Seul petit bémol… l'esthétique d'affichage de ce champ dans un formulaire est discutable.

Placer le curseur à la ligne devant laquelle vous désirez insérer un nouveau champ d'enregistrement avec une liste de choix. Appelez le menu Insertion/Champ de recherche.

Remarque concernant les champs mémo: Dès que vous faites certains types des requêtes contenant des champs mémo, ceux-ci sont tronqué à 256 caractères… Voici quelques solutions:

1. Lorsque vous avez une requête comportant un champ mémo avec la clause **Groupe**, le mémo sera tronqué il faut mettre sur ce champ mémo (en particulier!) avec la clause **Premier** et laisser les autres avec **Groupe**.

2. Si vous faites une requête qui renvoie les lignes **Uniques** le champ mémo sera tronqué. Il faut alors l'exclure de cette requête et le ramener plus tard en faisant une requête de la requête pour le rapatrier via les clés primaires.

3. Si vous utilisez une requête d'**Union**, le champ mémo sera tronqué. Dans le code **SQL** de la requête remplacez **Union** par **Union All**.

Ces limitations proviennent du moteur JET (qui n'est plus le moteur JET depuis MS Access 2007…). C'est un compromis de performance!

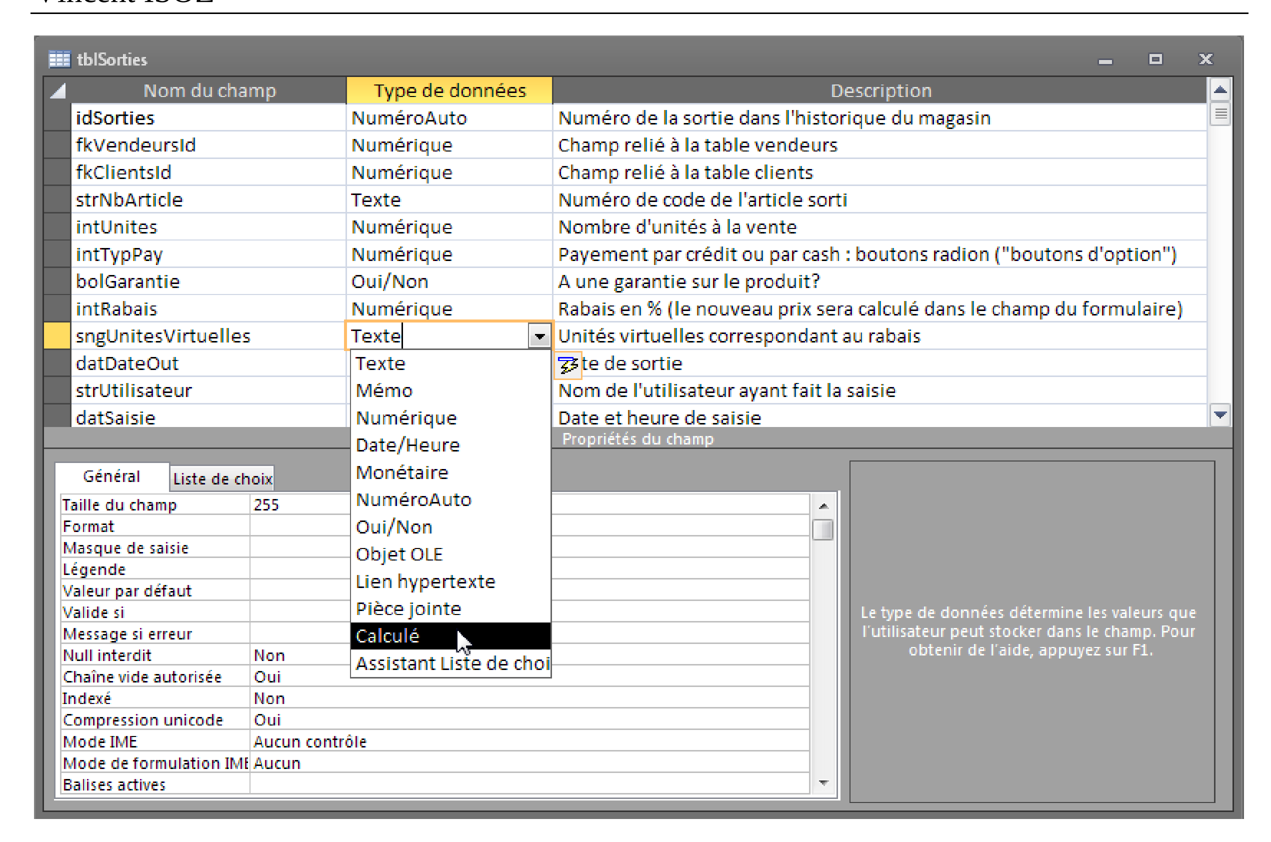
Calculé - Calculated

Nouveautés depuis MS Access 2010, permet de créer des calculs directement dans les tables ce qui est fort pratique si l'on reste dans le monde MS Access mais dès qu'il s'agit de migrer ensuite vers une autre technologie, c'est la catastrophe à gérer.

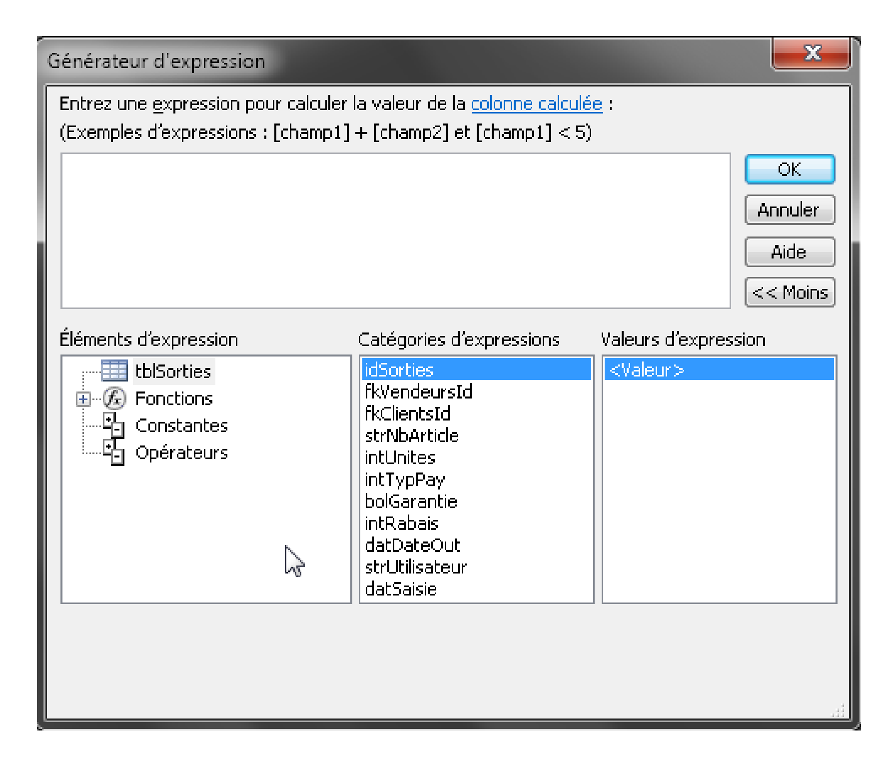
Voyons un exemple (idiot) de cette nouveauté... Considérons que quelqu'un souhaite dans le cas de ventes dans un magasin avoir dans sa table sortie (*tblSorties*) les unités de ventes correspondant au rabais effectué sur la commande de l'article en question. Nous nous baserons sur la propriété de commutativité de la multiplication:

$$P_{rabais} = \text{Unités} \cdot \text{PrixUnité} \cdot \left(1 - t\%\right) = \underbrace{\text{Unités} \cdot \left(1 - t\%\right)}_{\substack{\text{unités virtuelles} \\ \text{correspondant au rabais}}} \cdot \text{PrixUnité}$$

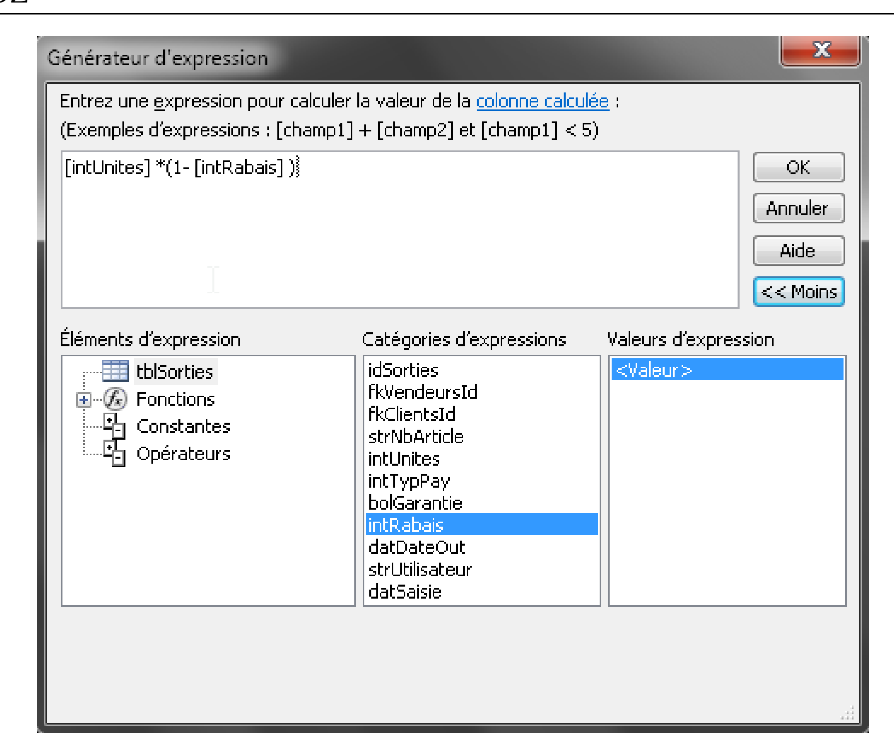
Nous ouvrons la table *tblSorties* en prenant soin de créer un nouveau champ appelé *sngUnitesVirtuelles*:



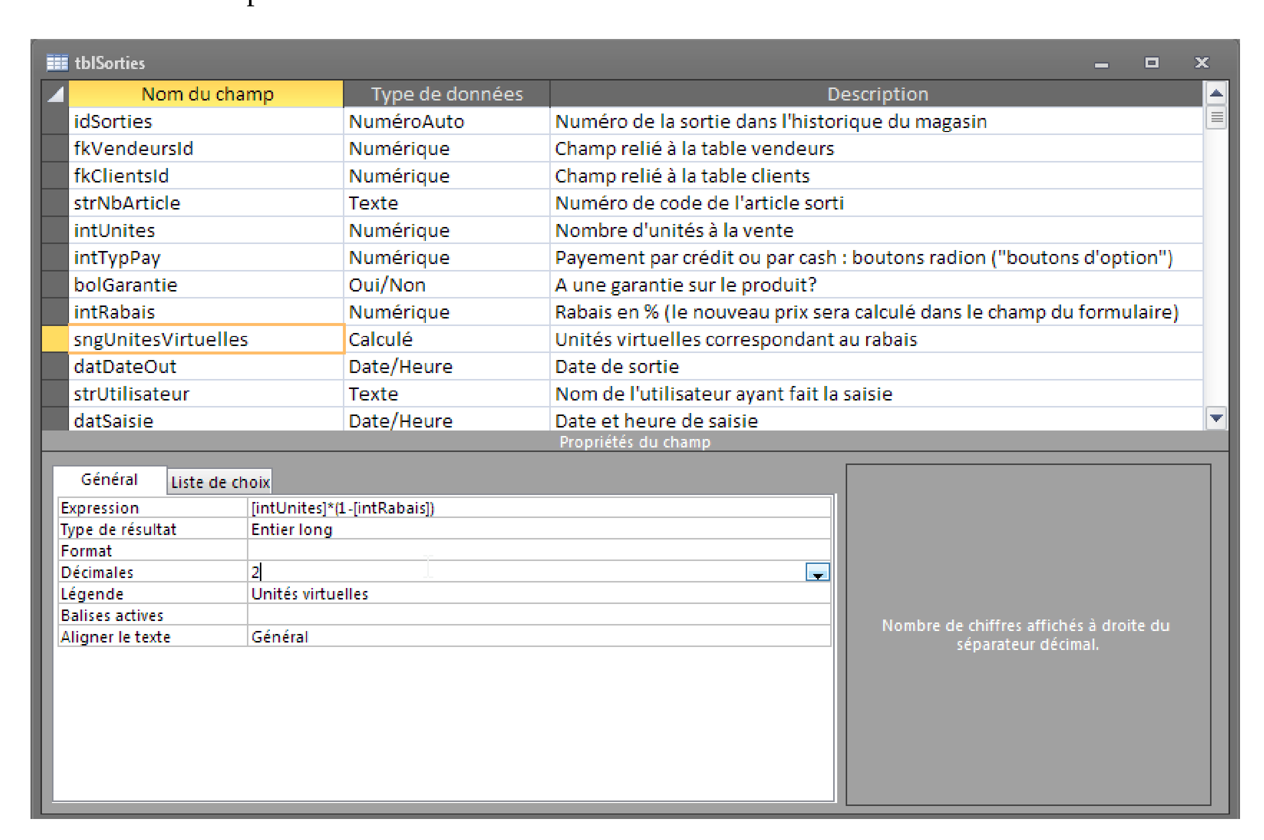
Dès la sélection de l'option *Calculé* effectuée, la boîte de dialogue suivante servant d'assistant (mais ceux qui n'ont pas envie de l'assistant peuvent écrire directement les formules de mémoire):



Comme ce "nouveau champ" interdit l'utilisation des fonctions autres que les calculs arithmétiques élémentaires, pour revenir à nos moutons.... nous faisons alors une bête formule d'arithmétique élémentaire:

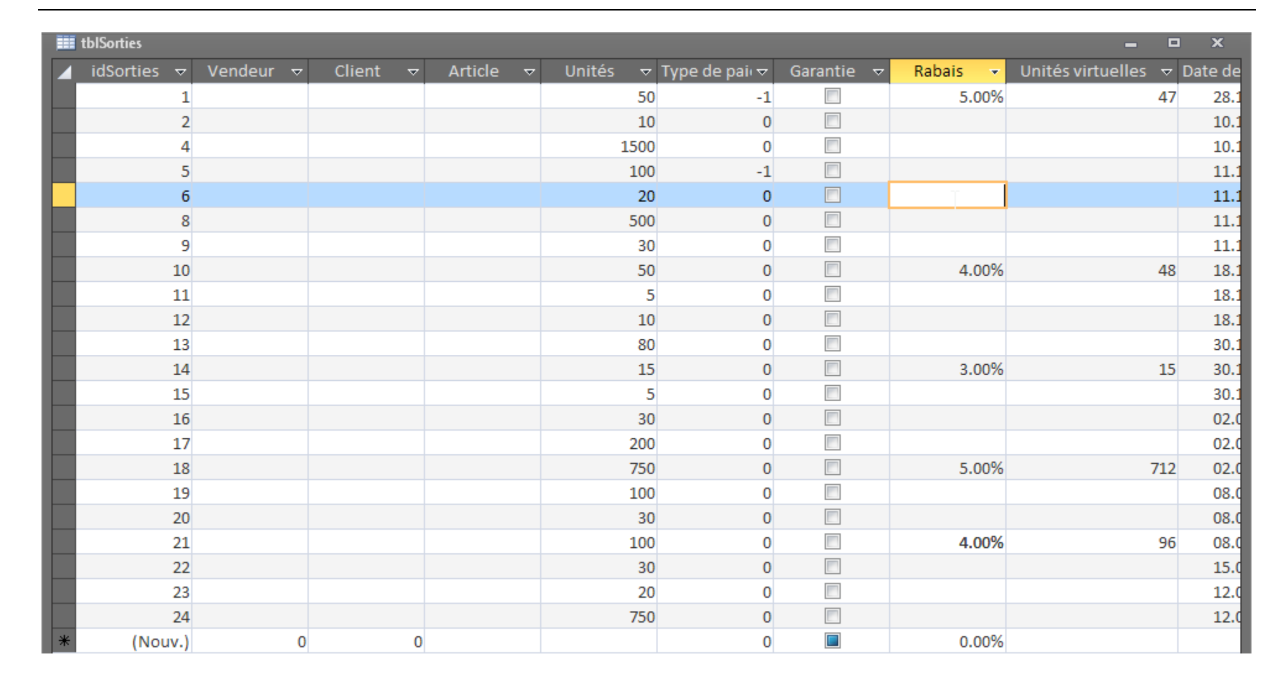


et nous validons par *OK*. Nous avons alors:



où le lecteur aura constaté que dans l'onglet *Général* et plus particulièrement dans le champ *Expression* qui s'y trouve, notre formule a été reportée par l'assistant et donc in extenso il est possible d'écrire directement aussi des formules dans ce champ.

Ensuite, si nous passons en affichage *Feuille de données* en ayant pris soin d'enregistrer les modifications au prélable, nous avons alors:



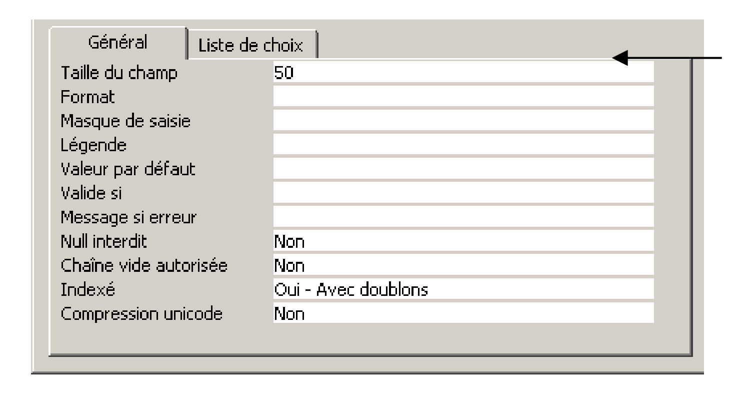
tblSorties

idSorties	Vendeur	Client	Article	Unités	Type de pai	Garantie	Rabais	Unités virtuelles	Date de
1				50	-1	☐	5.00%	47	28.1
2				10	0	☐			10.1
4				1500	0	☐			10.1
5				100	-1	☐			11.1
6				20	0	☐			11.1
8				500	0	☐			11.1
9				30	0	☐			11.1
10				50	0	☐	4.00%	48	18.1
11				5	0	☐			18.1
12				10	0	☐			18.1
13				80	0	☐			30.1
14				15	0	☐	3.00%	15	30.1
15				5	0	☐			30.1
16				30	0	☐			02.0
17				200	0	☐			02.0
18				750	0	☐	5.00%	712	02.0
19				100	0	☐			08.0
20				30	0	☐			08.0
21				100	0	☐	4.00%	96	08.0
22				30	0	☐			15.0
23				20	0	☐			12.0
24				750	0	☐			12.0
(Nouv.)	0	0			0	☐	0.00%		

Donc toutes les anciennes données ont automatiquement la formule qui a été calculée et toutes les nouvelles lignes ajoutées auront automatiquement le calcul qui s'exécutera lors de la saise des informations nécessaires à cela.

4.4.3 Formats

Pour chaque type de donnée, vous pouvez spécifier en plus un format dans la zone prévue à cet effet:



Il faut définir ces formats avant de commencer à saisir des valeurs dans la table (le principe est cependant semblable à MS Excel) !!!

- #.##0" Kg"
- #.##0 " kg réserve";-#.##0 "Kg achat"[Rouge]
- jjjj (jours), mmmm (mois), aaaa (année), t (trimestre), ss (semaine)
- hh (heures), m (minutes), s (secondes)
- #'###.00;-#'###.00[Rouge];0'000.00;"Non défini"

- > met toutes les données en majuscules (mais à éviter car ce n'est qu'une couche visuelle!)

Remarques:

R1. Ces options peuvent êtres définies aussi sur des champs de formulaires en allant dans les propriétés de ceux-ci.

R2. Le format de comptabilité "CHF " * #'##0.00 avec l'étoile pour avoir le symbole monétaire à gauche et la valeur à droite ne fonctionne plus depuis Access 2007… (sic!).

4.4.4 Masques de saisie

Par la suite, lorsque l'on précise qu'une entrée est obligatoire, cela signifie que si rien n'est saisi à l'endroit défini par le symbole donné, alors MS Access retourne un message d'erreur à l'écran.

Voici le code propre aux masques de saisie pour MS Access (ces codes peuvent se retrouver dans l'aide d'Access très facilement) qu'il est recommandé d'utiliser **que dans des champs de type Texte ou Date (à moins que l'on souhaite se compliquer la vie!!!):**

0 Chiffre (0 à 9, entrée obligatoire d'un chiffre, signes (+) et moins (-) non acceptés).

9 Chiffre ou espace (entrée facultative, signes plus et moins non acceptés).

& Caractère générique pour une lettre ou un espace

Chiffre ou espace (entrée facultative, positions vierges converties en espaces en mode édition, mais les espaces sont effacés lors de la sauvegarde des données, signes plus et moins acceptés).

L Lettre (A à Z minuscule ou majuscule, entrée obligatoire d'une lettre).

? Caractère quelconque ou espace (entrée obligatoire).

C Caractère quelconque ou espace (entrée facultative).

. ,: ; - / Séparateur de décimales, de milliers, de date et d'heure (à loisir…)

A Lettre ou chiffre (entrée obligatoire d'une lettre ou d'un chiffre).
Attention le nombre de *A* ne limitera cependant pas la taille d'un chiffre si le champ est de type *Numérique*. De plus vous aurez des surprises si vous utiliser les maques avec les réels simples.

a Lettre ou chiffre (entrée facultative).
Attention le nombre de *A* ne limitera cependant pas la taille d'un chiffre si le champ est de type *Numérique*. De plus vous aurez des surprises si vous utiliser les maques avec les réels simples.

<	Tous les caractères qui suivent sont transformés en minuscules
>	Tous les caractères qui suivent sont transformés en majuscules
\	Échappement d'un caractère (le caractère qui suit devra s'afficher normalement)

Tableau 3 Codes pour masque de saisie

Remarque: Si vous affectez à la propriété *Masque de Saisie* la valeur "Mot de passe", vous créez une zone de texte permettant de saisir un mot de passe. Tous les caractères tapés dans la zone sont enregistrés mais remplacés à l'écran par un astérisque (*). Vous utilisez le masque de saisie Mot de passe pour empêcher l'affichage des caractères tapés.

Exemples:

(0) 00-00-00-00	→	(1) 55-50-24-48
(9) 99-99-99-99	→	(1) 55-50-24-48
	→	( ) 55-50-24-48
(0) AA-AA-AA-AA	→	(1) 55-55-TE-LE
#999	→	–20
	→	2000
>L????L?0005L0	→	VERTEVE339M3
	→	MAI R 452B7
>L0L 0L0	→	T2F 8M4
00000-9999	→	98115-
	→	98115-3007
>L<??????????????	→	Marie
	→	Dupont
SSN 000-00-0000	→	SSN 555-55-5555
>LL00000-0000	→	DB51392-0493

Sous forme finale:

Tableau 4 Codes de masque de saisie types

Définition du masque	Avant saisie	Exemple de saisie
00\ 00\ 00\ 00\ 00;0;_	__ __ __ __ __	01 23 45 67 89
LLL"--"??;;@	@@@--@@	aze--r
\(000") "000\-0000;1;*	(***) ***-****	(206) 555-0248
>L<CCCCCCCCCCCCCC	_______________	Jean-claude sohm
"ISBN "0\-&&&&&&&&&\-0	"ISBN "_-_____________-_	ISBN 5-126795111-8
####	____	-26
+41(0)00\/000.00.00	+41(_)__/___.__.__	+41(0)76/329.53.88

Spécifications sur les masques de saisie, en considérant l'exemple:

\(000") "000\-0000;1;*

le "1" après le premier ";" signifie" qu'il faudra stocker dans la table seulement les données saisies (alors que la valeur "0" permet d'indiquer de stocker tous les caractères visibles). La valeur après le deuxième ";" représente le caractère utilisé pour représenter l'espace réservé à chaque caractère à saisir.

La principale différence entre la définition d'un style de saisie dans le champ "format" ou dans le "masque de saisie" est la suivante:

Si vous définissez un style dans le champ "Format", alors la saisie sera adaptée lorsqu'elle sera insérée dans la table et non pas au moment même de la saisie dans le formulaire correspondant (ce qui est le rôle du masque de saisie).

Remarque:

R1. Les données du masque, bien qu'elles apparaissent dans les tables, ne s'y trouvent pas physiquement. Ainsi, si vous exportez une table avec un masque vers MS Excel, vous perdrez tous les caractères du masque.

R2. Ces options peuvent être définies aussi sur des champs de formulaires en allant dans les propriétés de ceux-ci.

4.4.5 Validation (Valide Si)

Les options "Valide Si" et "Message si erreur", ont pour fonction d'afficher un message d'erreur personnalisé si le champ saisi ne correspond pas à des données ayant été définies par des relations d'ordre (ce qui n'est pas toujours possible) ou dans un langage plus commun: par des critères.

Voici quelques exemples:

1. N'autorise que la saisie d'une valeur supérieure à 5:
   >=5

2. N'autorise la saisie d'une valeur qu'inférieur à la date spécifiée:

   <#12.01.93#

3. N'autorise seulement que les textes commençant ou plus grand que la lettre spécifiée:

   >"A"

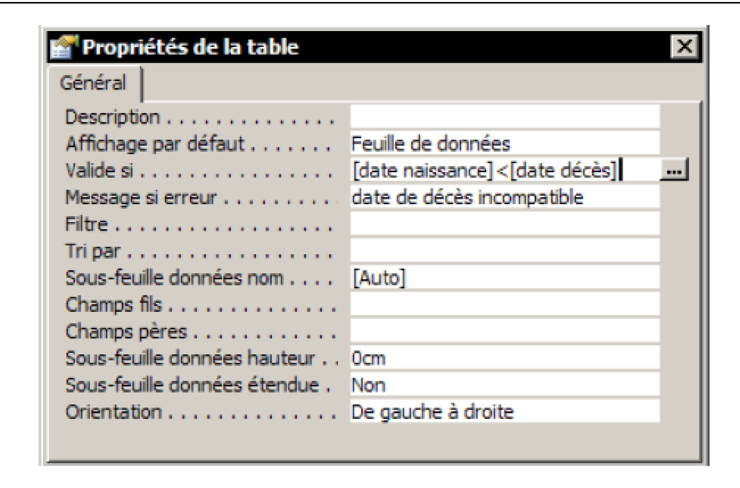
4. N'autorise que la saisie de valeurs égales à celle spécifiées:

   ="Madame" OU ="Monsieur"

5. N'autorise que la saisie d'une date comprise entre la date de jour et la même date mais 65 ans plus tôt:

   =SérieDate(Année (Maintenant ())-65;Mois(Maintenant());Jour(Maintenant())

Il est aussi possible de mettre une règle de validation au niveau de la table. Effectivement, lorsque vous êtes en *Mode création* et que vous cliquez sur le bouton apparaît la boîte de dialogue suivante:



dans laquelle nous voyons un exemple d'application où la règle impose que l'information (date) se trouvant dans le champ nommé *date naissance* est inférieure à la date se trouvant dans le champ *date décès*.

Attention!!!! Des critères multiples dans le *Valide Si* nécessitent obligatoirement l'usage de la fonction VRAIFAUX( ) qui est l'équivalent de la fonction SI( ) de MS Office Excel

Maintenant que vous avez parcouru cette liste non exhaustive, nous allons passer à un petit exercice pratique:

Ouvrez la table *tblArticles* et entrez pour les noms de champs *N° Article*, *N° Fournisseur* et *Quantité par unité de commande* une description comme ci-dessous (si l'équivalent n'a pas déjà été fait):

Articles : Table

Nom du champ	Type de données	Description
N° Article	Texte	N° Article à 7 positions
Désignation	Texte	
N° Fournisseur	Numérique	Résulte de la table "Fournisseurs"
Quantité par unité de comma	Numérique	Quantité par unité auprès du fournisseur
Prix unitaire	Monétaire	

Modifiez la structure de la table en:

N° Article
Taille du champ = 7 et Nul interdit

Désignation
Taille du champ = 30 et Nul interdit

N° Fournisseur
Format Nombre général (Entier) Décimales 0

Quantité par unité de commande
Valeur par défaut 5 Validité si >=5 (la logique de lecture est inverse de ce que l'on a l'habitude d'utiliser) et message si erreur 'Valeur trop petite'

Prix unitaire
Format: Monétaire et valeur par défaut 0

Fermez et enregistrez la table et testez ensuite le formulaire instantané que nous avions créé en tout début de cours pour voir comment il se comporte maintenant que nous avons défini toutes ces propriétés (il faut prendre l'habitude de faire systématiquement ce genre de contrôles).

Définissez pour le champ d'enregistrements *N° Article*, un *masque de saisie* qui n'accepte pour l'entrée au trois premières positions que trois lettres et pour les trois dernières, trois nombres séparés par un trait d'union (>LLL-000).

Etablissez à l'aide de l'assistant de liste de choix, une liste telle que l'on puisse choisir le *N° Fournisseur* dans la table *Articles* au lieu de le saisir (cette liste doit afficher également les noms des fournisseurs et s'adapter automatiquement si des nouveaux sont entrés dans le table *Fournisseurs*). Changez pour le *Mode feuille de données* et vérifiez le résultat.

Ouvrez l'ancien formulaire de la table *tblArticles* et testez la saisie. Comme vous pouvez certainement le voir, le formulaire ne s'est pas adapté aux nouvelles propriétés des champs. Deux possibilités s'offrent alors à vous:

1. Ecraser l'ancien formulaire instantané par un nouveau (option très gênante si vous avez passé beaucoup de temps à faire des modifications sur ce premier)

2. Mettre à jour l'ancien formulaire en supprimant les champs qui ont changés en cliquant sur le bouton suivant de la barre d'outils (quand votre formulaire est en mode création):

Ce bouton va faire apparaître une petite fenêtre dans laquelle se trouvent tous les champs utilisés par votre formulaire. Il suffit ensuite de faire un "glisser/déplacer".

Créez un formulaire instantané pour la table *tblFournisseurs* et insérez un nouveau fournisseur (laissez jouer votre imagination). Enregistrez ce formulaire sous le nom "Fournisseurs".

Revenez dans le formulaire *Articles* et testez si le numéro du nouveau fournisseur apparaît dans la liste des numéros de fournisseurs.

Changez le champ d'enregistrement *NPA* de la table *tblFournisseurs* en un champ d'enregistrement de type "Texte" avec une taille de 5 caractères.

Renommez le champ d'enregistrement *Délais de livraison/Jours* en *Délais de livraison* et définissez un format d'affichage qui contienne le texte "jour(s)". (#" jour(s)"). Définissez la valeur par défaut à 0.

Dans la table *tblSorties* définissez la valeur par défaut du champ d'enregistrement *Date de Sortie* comme étant la date du jour (fonction *Date*( )).

Ajouter dans la table *tblVendeurs* la colonne de *Provisions* (avec le type de données *Numériques,* au format *Pourcentage* avec taille de champ de type *Réel simple*) où seul des pourcentages avec au plus deux décimales après la virgule doivent pouvoir être saisis.

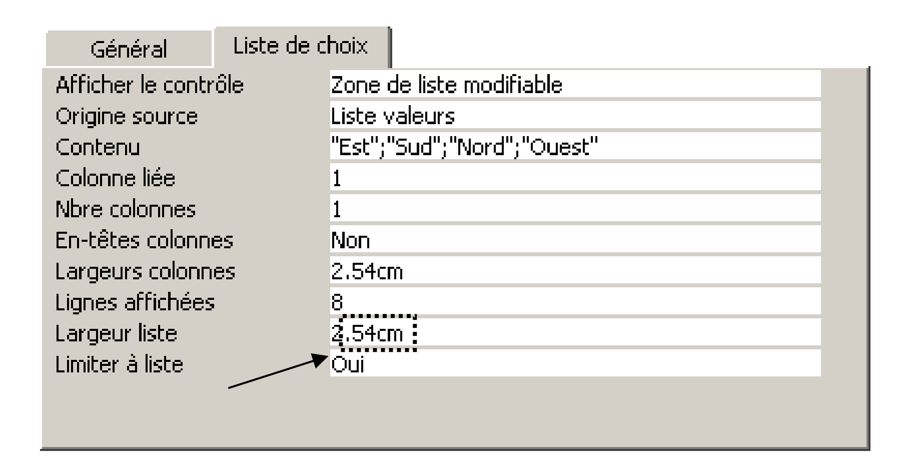
Saisissez les valeurs comme représentées sur la figure suivante:

Vendeurs : Table

	N°	Nom Vendeur	Provision
	1	Weidmann, Jean-Marc	3.00%
	2	Butty, Joe	3.50%
	3	Clerc, Marlène	5.00%
▶	4	Castrini, Rodolfo	10.00%
	5	De Siebental, Wille	5.00%
	6	Massa, Sybille	3.50%
	7	Mettraz, Sébastien	5.00%
	8	Barbier, Aline	5.00%
	9	Mettraux, Théo	3.50%
	10	Hoffmann, Thérèse	4.00%
	11	Mouther, Jean	3.00%
*	(NuméroAuto)		0.00%

Comme vous pouvez l'observer nous devons faire face à un problème relatif aux pourcentages. Parlez-en avec votre formateur.

Définissez pour le champ *Région* un liste de choix dans laquelle vous mettez à disposition les valeurs *Nord*, *Est*, *Sud*, *Ouest*. De plus, il faut empêcher l'utilisateur de pouvoir saisir autre chose que proposé dans cette liste. Ainsi:



Général | Liste de choix

Afficher le contrôle	Zone de liste modifiable
Origine source	Liste valeurs
Contenu	"Est";"Sud";"Nord";"Ouest"
Colonne liée	1
Nbre colonnes	1
En-têtes colonnes	Non
Largeurs colonnes	2.54cm
Lignes affichées	8
Largeur liste	2.54cm
Limiter à liste	Oui

Cette propriété est assez importante. Il faut essayer de ne jamais l'oublier. Ainsi, dans notre magasin, toutes les listes doivent être limitées.

Ouvrez l'ancien formulaire instantané de la table *Vendeurs* vérifiez si la liste fonctionne et au besoin mettez le champ à jour.

Insérez, toujours dans la table *Sorties*, un champ nommé "Payement en espèces" et un autre nommé "Payement par crédit". Pour chacun de ces champs, prenez comme type de données celui nommé "Oui/Non".

Créez un formulaire instantané pour la table *Sorties* et définissez au hasard, pour chaque enregistrement, le type de payement.

Evidemment on peut actuellement cocher les deux cases mais on verra plus tard avec les macros comment gérer ce genre d'absurdités.

4.5 Import MS Excel

Importez le fichier MS Excel *TableSorties.xls*. Le résultat doit être présenté comme ci-dessous (la clé primaire doit être sur le champ *N°*). Le but étant de connaître les entrées/sorties du magasin (on va créer la table des Entrées de suite après). Faites attention au fait que la colonne *N°* doit être de type "numérotation automatique" !

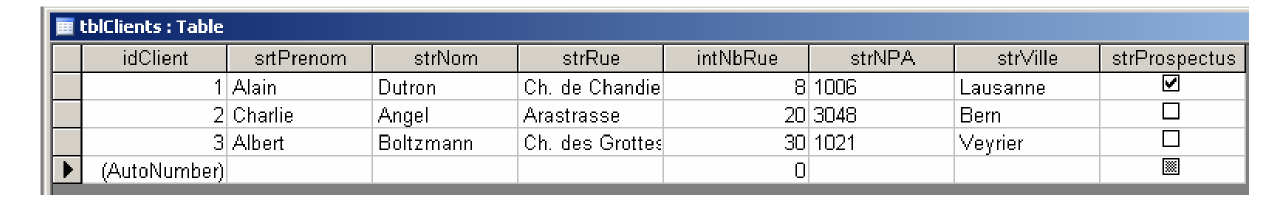
N°	N° Article	Nombre d'unités	Date de sortie
1	GEN-001	50	28.11.1996
2	GEN-003	10	10.12.1996
3	GEN-005	40	10.12.1996
4	INF-002	1500	10.12.1996
5	GEN-002	100	11.12.1996
6	GEN-003	20	11.12.1996
7	GEN-005	10	11.12.1996
8	INF-002	500	11.12.1996
9	INF-003	30	11.12.1996
10	GEN-002	50	18.12.1996
11	INF-001	5	18.12.1996
12	INF-003	10	18.12.1996
13	GEN-004	80	30.12.1996
14	INF-001	15	30.12.1996
15	INF-003	5	30.12.1996
16	GEN-003	30	02.01.1997
17	GEN-006	200	02.01.1997
18	INF-002	750	02.01.1997
19	GEN-001	100	08.01.1997
20	GEN-004	30	08.01.1997
21	INF-002	100	08.01.1997
22	GEN-003	30	15.01.1997
23	INF-002	20	12.02.1997
24	INF-004	750	12.02.1997

Créez manuellement (saisie clavier dans un formulaire instantané) la table *tblEntree* ci-dessous suivante (la clé primaire doit être sur le champ *idClient* pour l'instant).

N°	N° Article	Nombre d'unités	Date d'entrée
1	GEN-001	2	25.11.1996
2	GEN-006	3	26.11.1996
3	GEN-003	2	02.12.1996
4	GEN-005	1	03.12.1996
5	INF-003	2	03.12.1996
6	GEN-004	2	04.12.1996
7	GEN-002	4	05.12.1996
8	INF-002	2	06.12.1996
9	INF-003	5	06.12.1996
10	GEN-005	1	12.12.1996
11	INF-001	5	12.12.1996
12	GEN-002	4	20.12.1996
13	GEN-004	1	20.12.1996
14	INF-002	1	20.12.1996
15	GEN-004	1	17.01.1997
16	GEN-006	1	17.01.1997
17	INF-002	1	17.01.1997
18	INF-003	5	17.01.1997
19	INF-001	4	08.02.1997
20	INF-004	2	08.02.1997

Saisissez à partir de la fenêtre de base de données les propriétés des tables (bouton droit sur les tables). Fermez la base et observez ce qui ce passe (pas) !?

Créez manuellement (saisie clavier dans un formulaire instantané) la table *tblClients* ci-dessous (la clé primaire doit être sur le champ *idClient* pour l'instant).



tblClients : Table

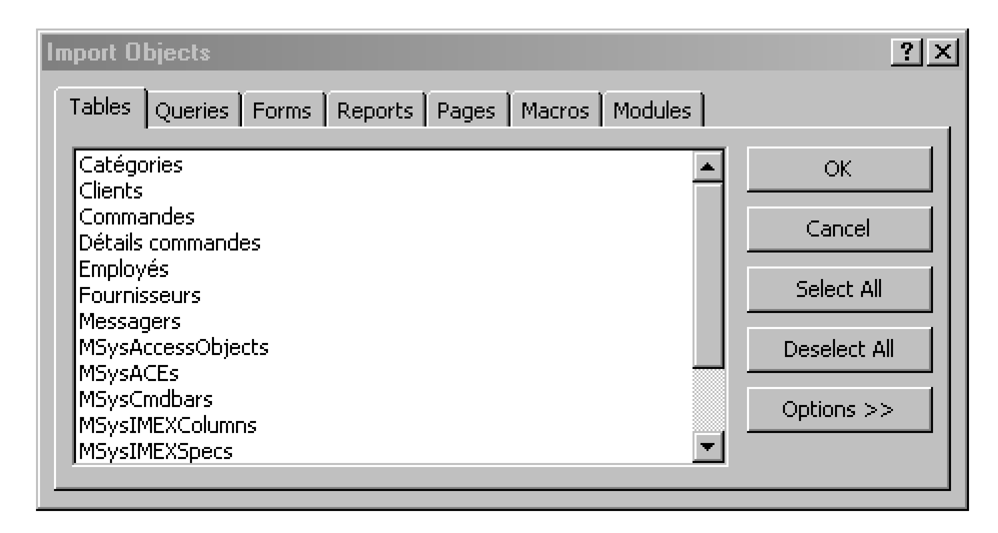
idClient	srtPrenom	strNom	strRue	intNbRue	strNPA	strVille	strProspectus
1	Alain	Dutron	Ch. de Chandie	8	1006	Lausanne	☑
2	Charlie	Angel	Arastrasse	20	3048	Bern	☐
3	Albert	Boltzmann	Ch. des Grottes	30	1021	Veyrier	☐
(AutoNumber)				0			

Dans la table *tblSorties*, rajoutez une colonne *ClientsId* qui permettra plus tard de créer des relations. Saisissez dans cette colonne des valeurs comprises entre 1 et 3 (y compris).

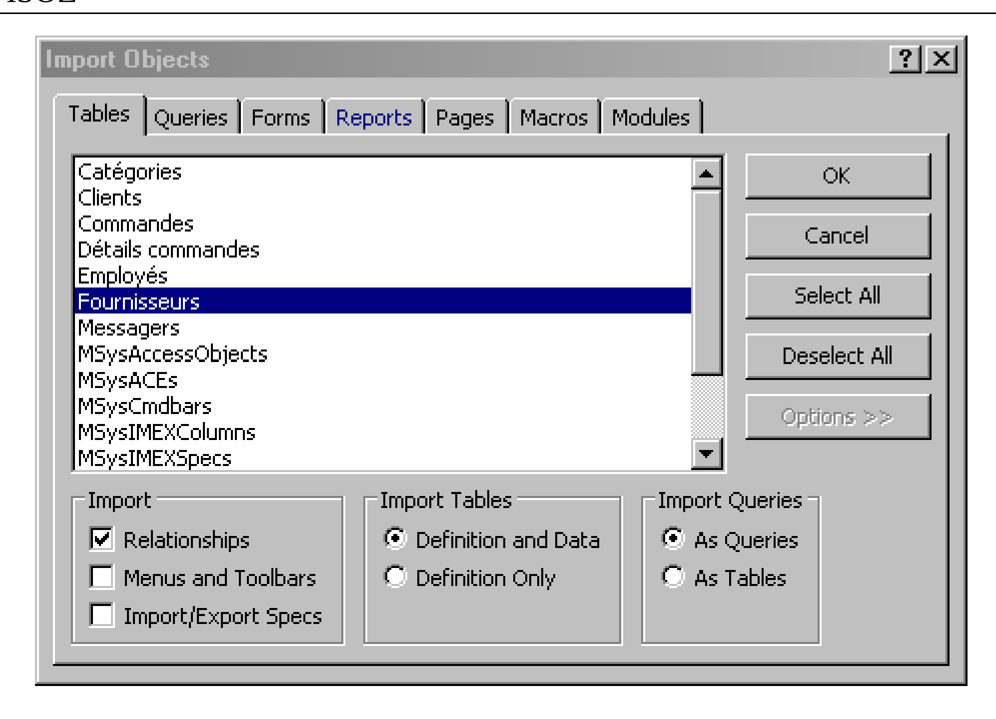
4.6 Import MS Access

Un cas relativement rare d'import et celui d'autres bases de données MS Access. De même que pour les tables MS Excel, il suffit d'aller dans le menu *Fichier/Import des données externes* et de choisir une des deux options.

Il faut savoir que l'option *Lier* ne permet que de lier des tables. Par contre, l'option *Import* permet d'importer tous les objets MS Access comme le montre la figure ci-dessous:

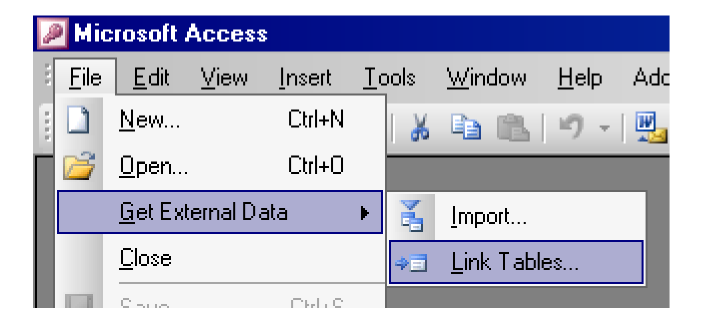


N'oubliez pas de cliquer sur le bouton *Options* qui vous donne des possibilités parfois pertinentes:

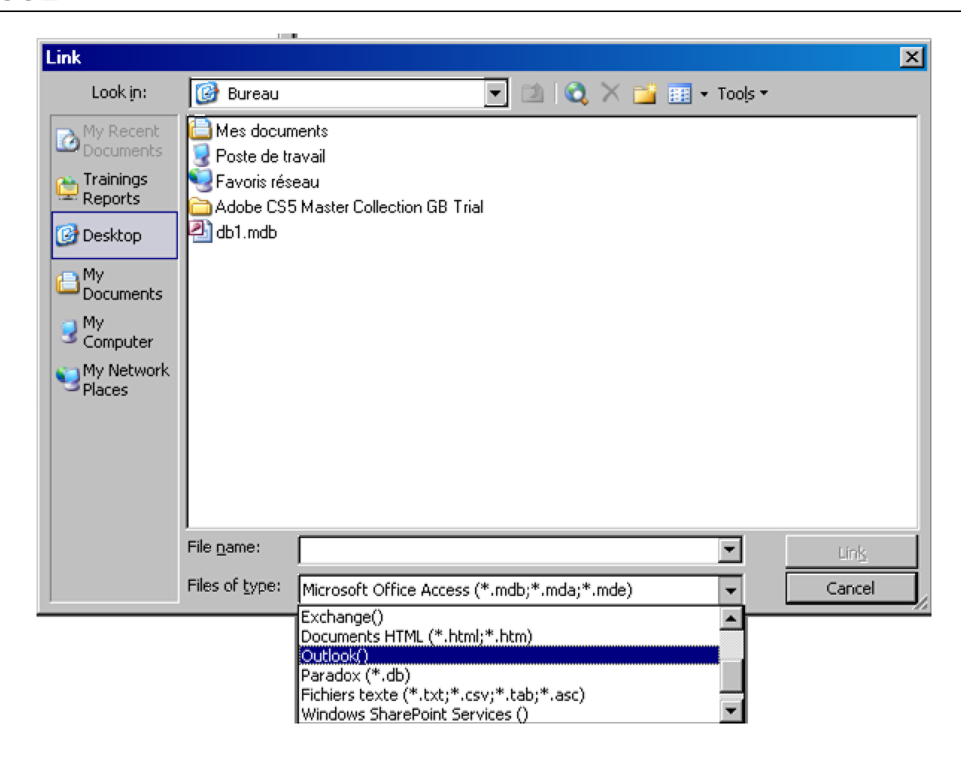


4.7 Import/liaison MS Outlook

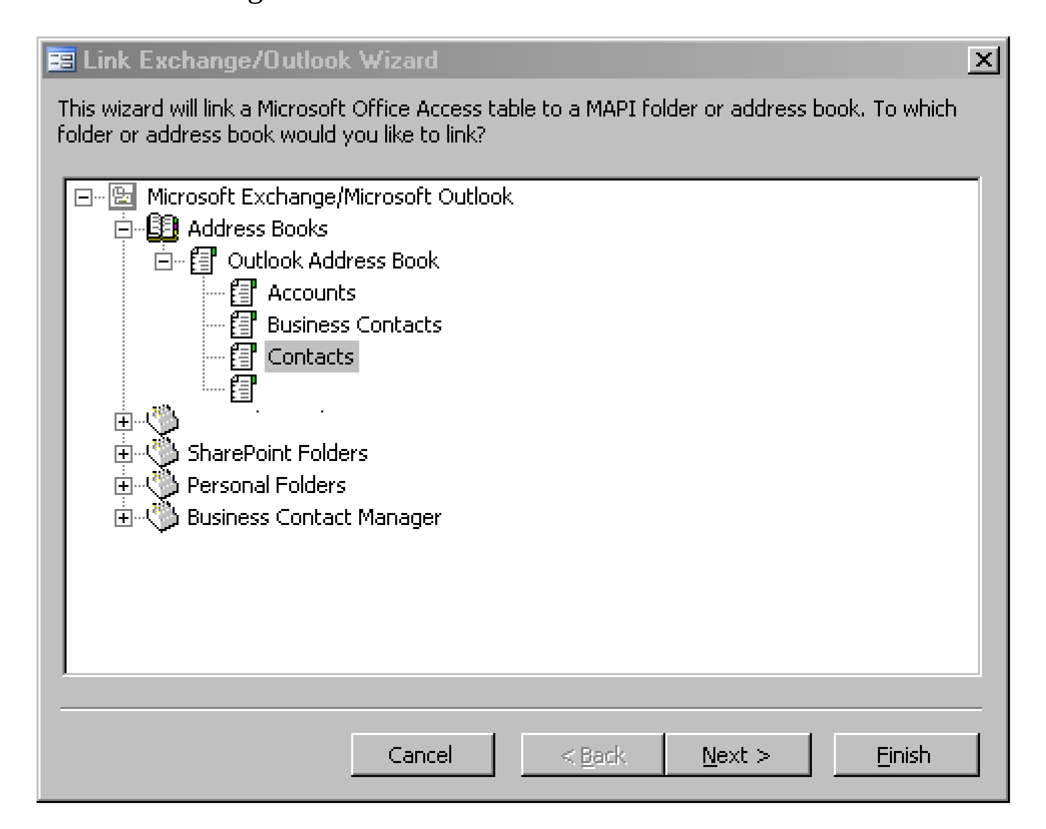
On peut se lier ou importer un carnet d'adresse d'entreprise ou local de MS Outlook (et également tout autre dossier Outlook!) avec une base de données Access. Le principe est le même que précédemment. Voyons comment faire:



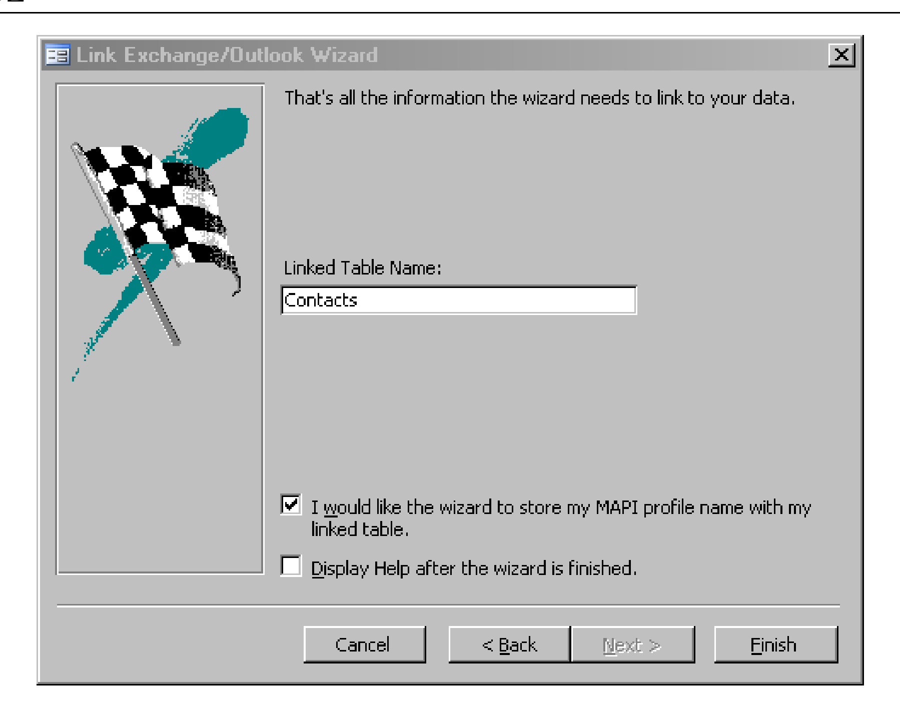
Vous allez donc dans *File/Get External Data/Link Tables*…et ensuite:



Ensuite, dans la liste des types de fichier, choisissez *Outlook*. Vient alors après quelques secondes la boîte de dialogue suivante:



Prenons le carnet d'adresse local *Contacts*. Validez par *Next* et vous aurez:

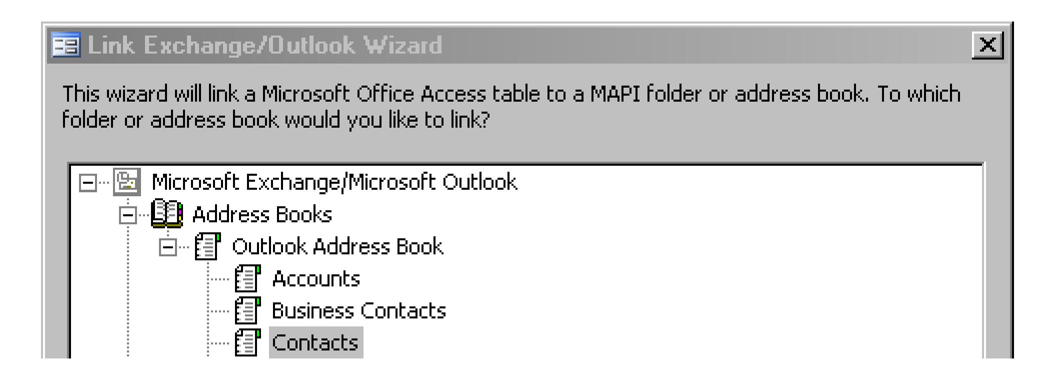


Validez par *Finish*. Et vous aurez:

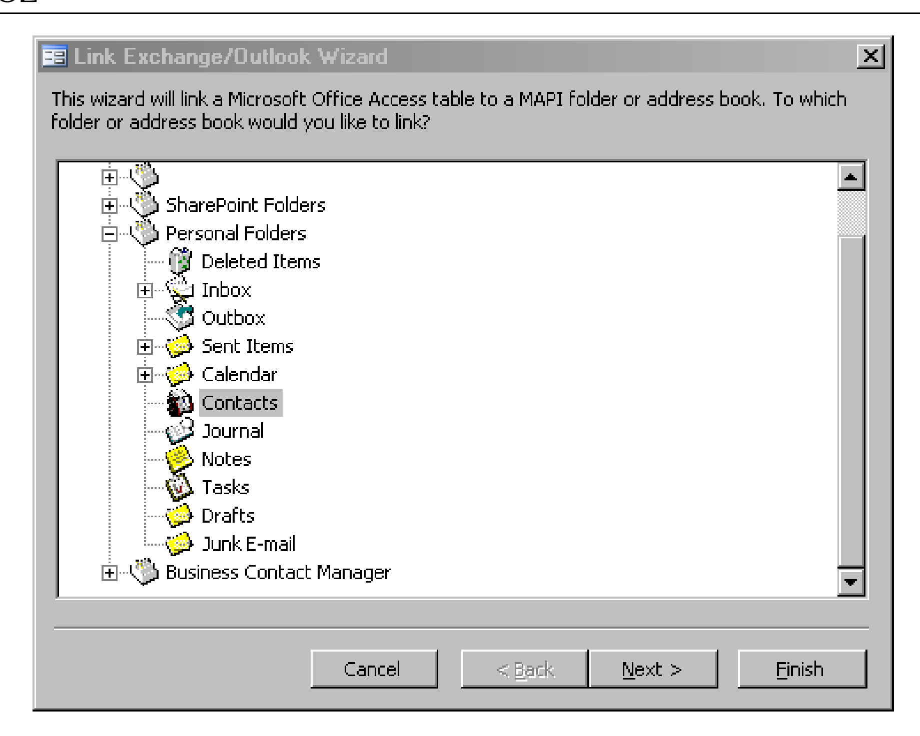


Mais cette méthode ne prend en compte que les modifications et suppressions d'éléments existants lors de la création de la liaison et elle est unidirectionnelle (c'est Outlook qui pilote Access!).

Il y a au fait un **piège** pour que la modification/ajout/suppression d'éléments soit pris en compte (mais toujours en unidirectionnel). Lors de l'assistant de liaison qui vous permet par exemple de choisir le carnet d'adresse:



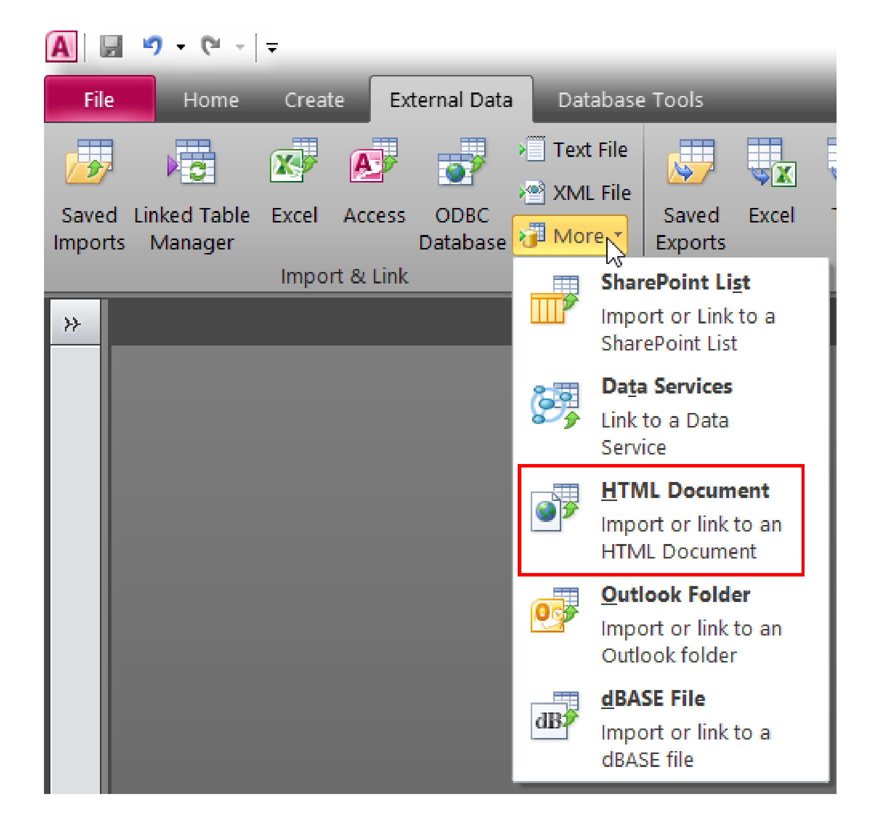
Il ne faut pas prendre celui-ci de la liste des *Adress Books* mais de la liste des dossiers du *Personnal Folders* si situant en-dessous:



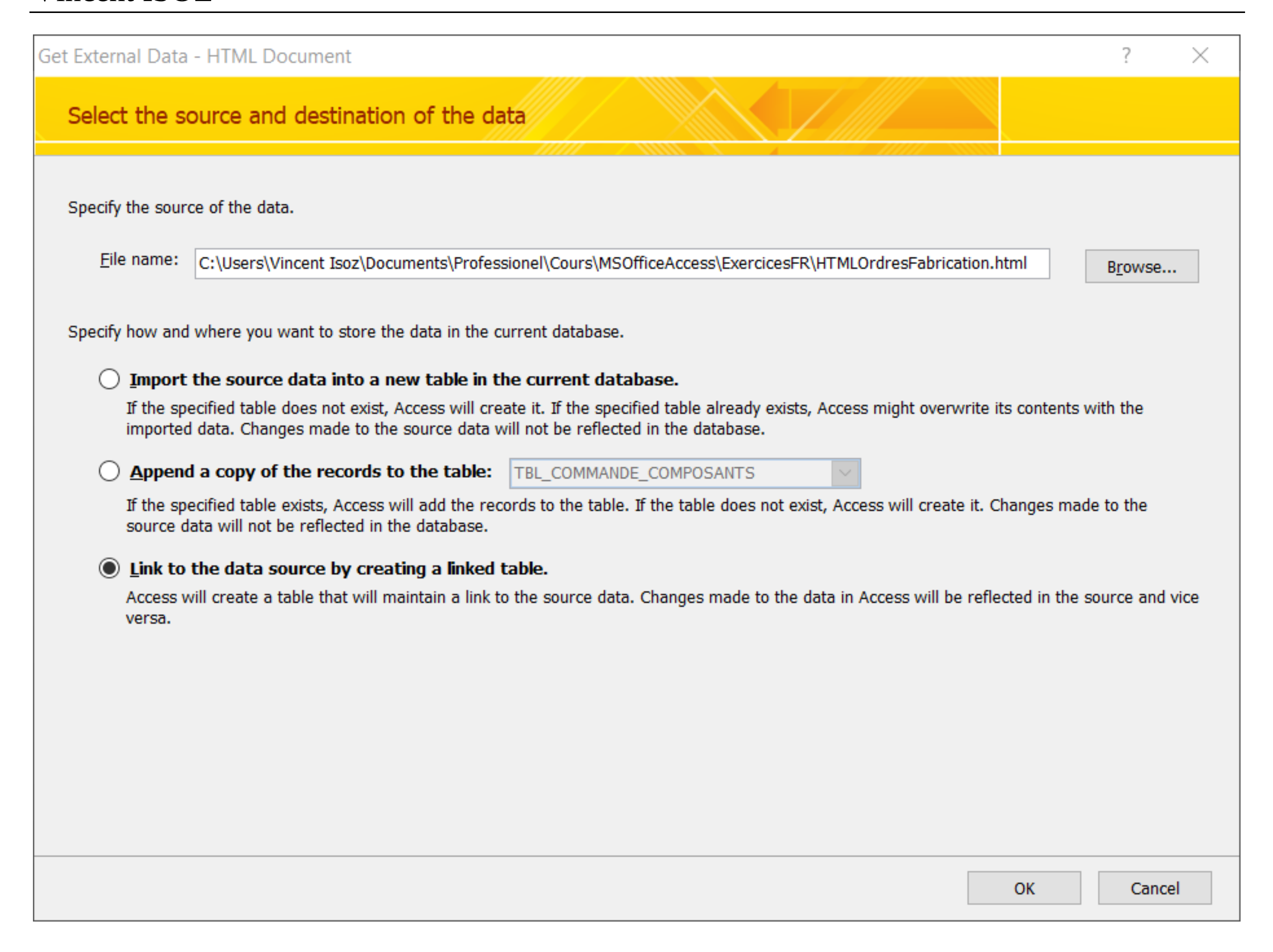
Après quoi cela marche (donc les modifications sont prises au moins en unidirectionnel…!).

4.8 Import HTML

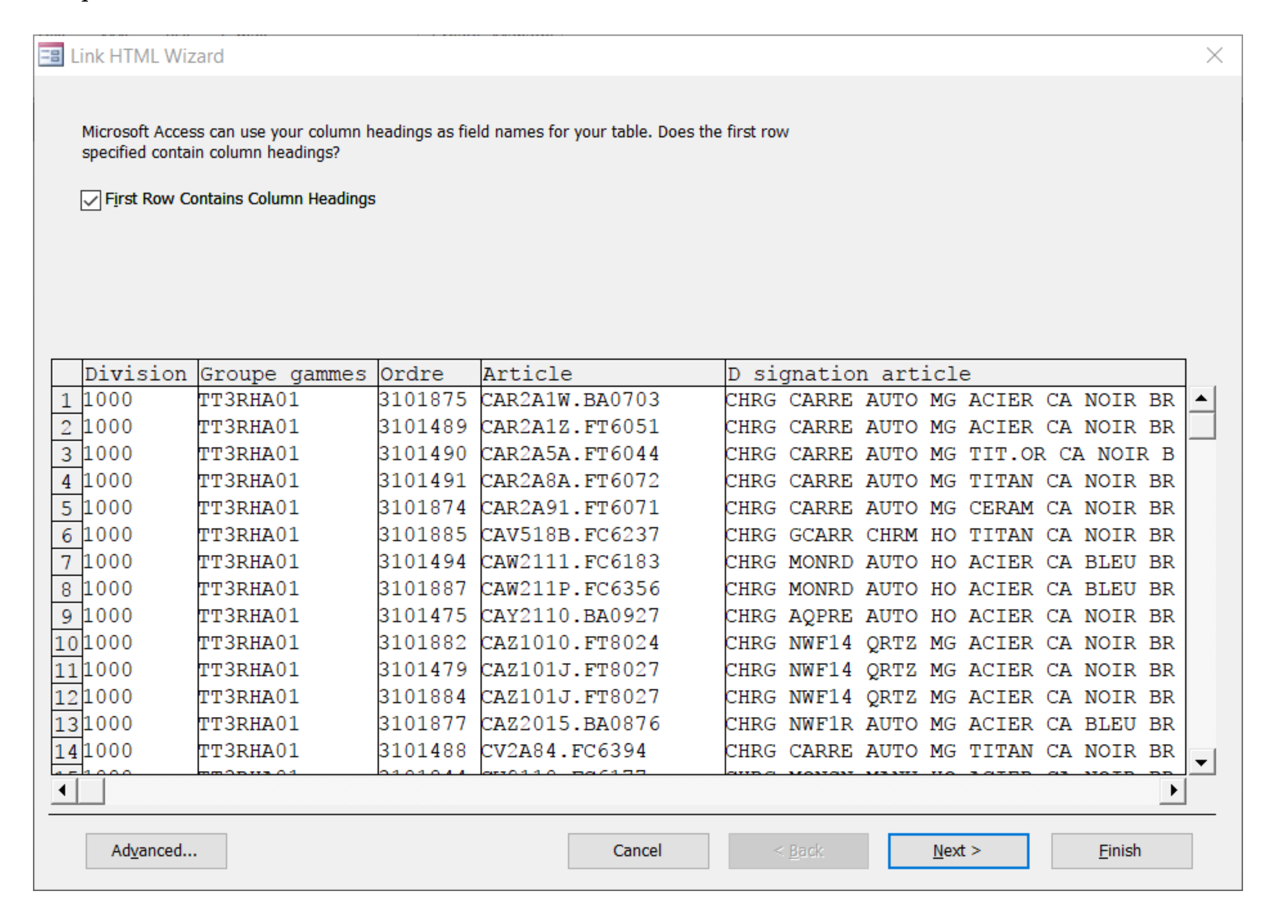
Pour importer de fichier HTML, nous passons par:



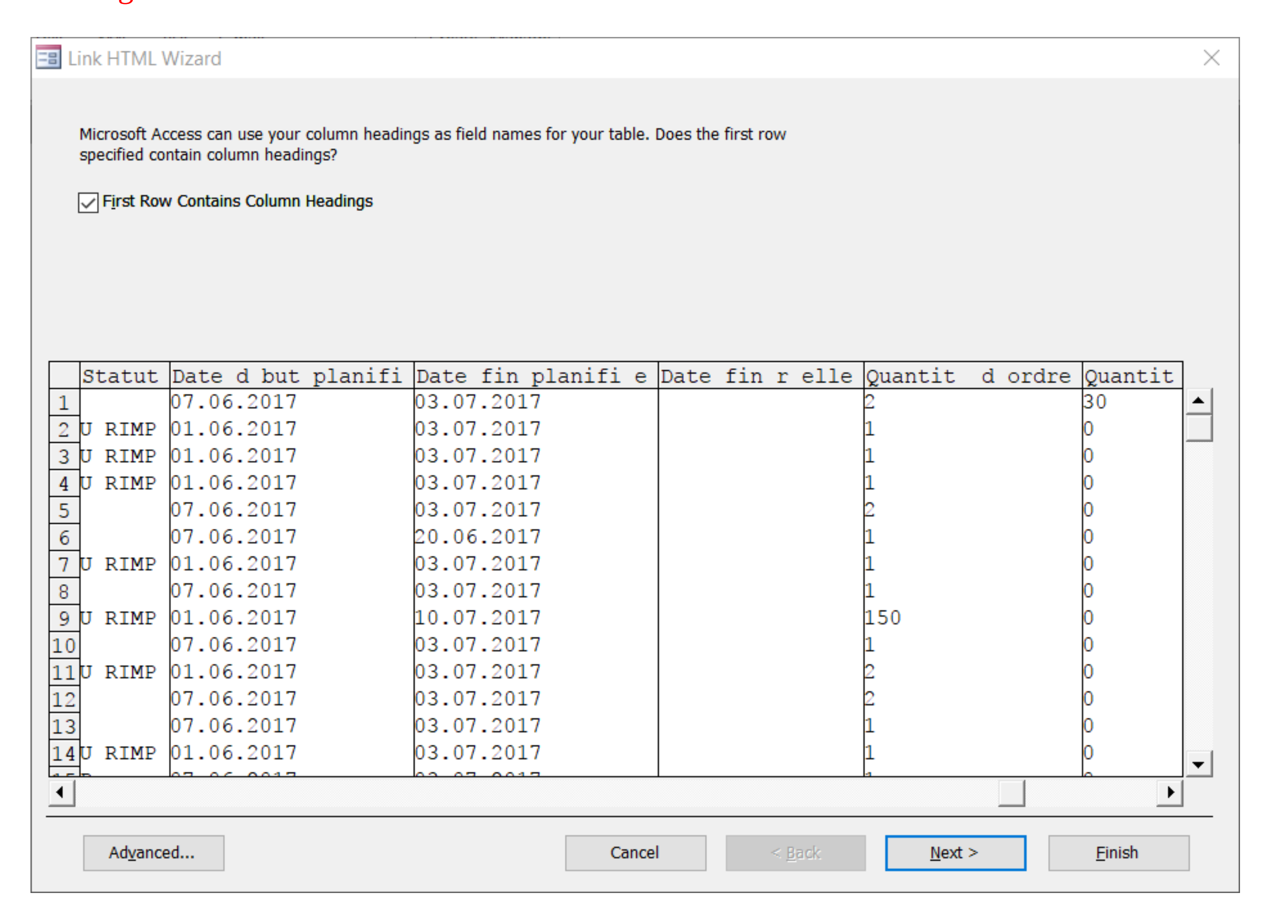
Nous avons alors l'assistant:



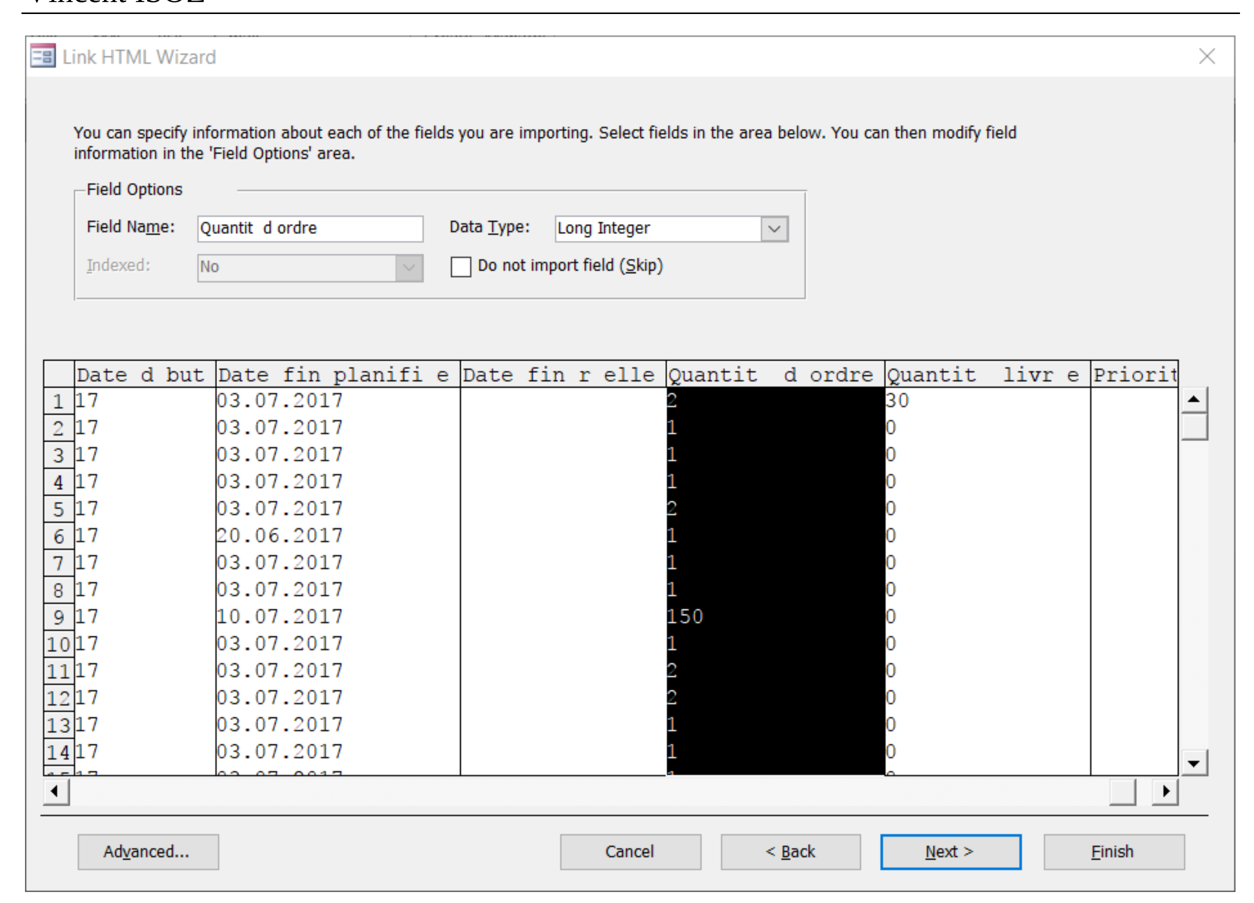
Cliquez sur *OK* et l'assistant continue:



Vous remarquerez que ce fichier HTML a des problèmes au niveau des accents et apostrophes de la ligne de titres:

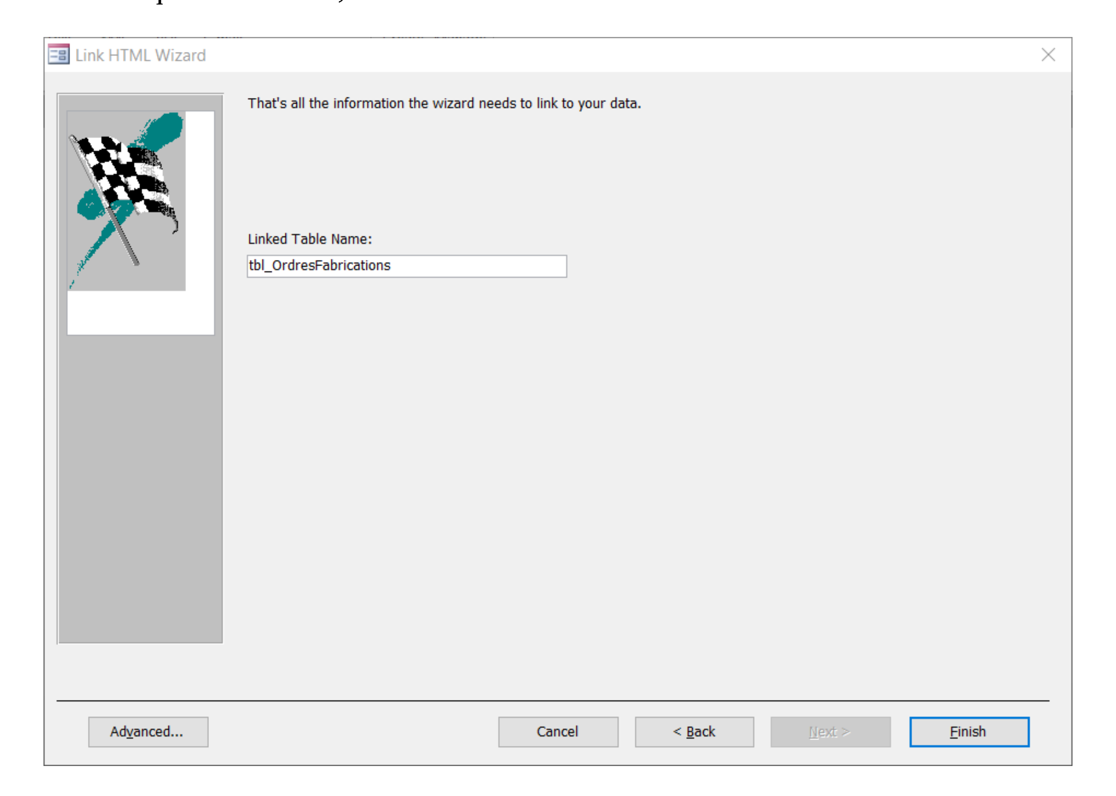


La prochaine étape de l'assistant vous permet de changer le nom des colonnes qui seraient éventuellement inadéquates:

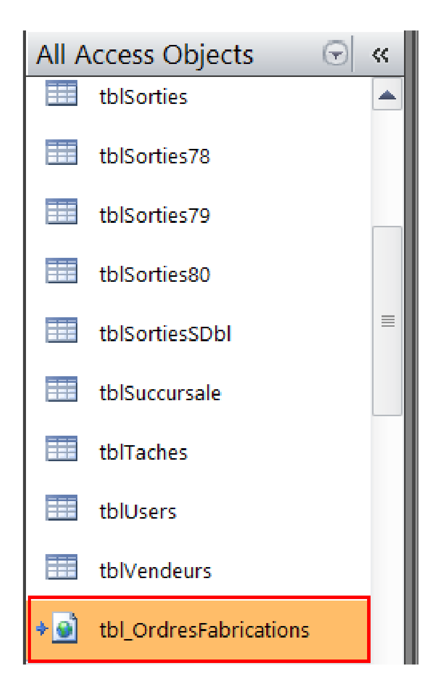


Mais ne le faites pas! Effectivement avec les fichiers HTML dont les accents ne sont pas reconnus car mal encodés, alors tout type de formule (typiquement DLookUp) que vous feriez avec les nouveaux noms ne marchera pas!!!!!!!!!!!!!

Quand nous cliquons sur *Next*, il nous demande le nom de la table:



Et si nous cliquons sur *Finish*:



4.9 Format des tables

Etablissez en mode de feuille de données, la table *Vendeurs* (*tblVendeurs*) avec les champs d'enregistrements *Nom du Vendeur* (*strVendeurs*) et *Région de Vente* (*StrRgVentes*). Entrez ensuite les données ci-dessous:

N°	Nom du Vendeur	Région des Ventes
1	Weidman, Jean Ma	Est
2	Butty, Joe	Sud
3	Clerc, Marlène	Nord
4	Castrini, Rodolfo	Nord
5	De Siebental, Willi	Est
6	Massa, Sybille	Ouest
7	Mettraz, Sébastien	Sud
8	Barbier, Aline	Nord
9	Mettraux, Théo	Est
10	Hoffman, Thérèse	Ouest
11	Mouther, Jean	Nord

Générez automatiquement la clé primaire lors de la fermeture de la table. Ouvrez de nouveau la table *tblVendeurs*, puis changez la désignation de la colonne *ID* en *idVendeur*.

En mode création vérifiez que les types de données soient adéquats.

Changez la désignation du champ *N°* de la table *tblSorties* en *idSortie*

Insérez entre les colonnes *N° Sortie* et *N° Article* de la table *tblSorties*, une nouvelle colonne *N° Vendeur* (*tblVendeurId)*. Dans cette colonne, le vendeur doit indiquer quelle vente il a réalisé en vue du calcul ultérieur des commissions (saisissez des valeurs aléatoires entre 1 et 10).

Déplacez la colonne *N° Vendeur* (*tblVendeurId*) au début de la table, puis annulez cette opération et déplacez la colonne *N° Vendeur* (*tblVendeurId*) à la fin de la table.

Figer les colonnes *N° Sortie* (*idSortie*) et *N° Article* (*idArticles*) en passant par *Format/Figer les colonnes*. Cette fonction se base sur le même principe que celle se trouvant dans MS Excel. Puis modifiez le layout de la table en masquant les colonnes *Nombre d'unités* (*intUnites*) et *Date de Sortie* (*datSortie*) (*Format/Masquer les colonnes*). Cette fonction est utile lorsque l'on a besoin d'imprimer seulement certaines informations d'une table.

Supprimer le figement des colonnes et donnez à la table *tblArticles*, le layout suivant (Access 2000 et ultérieur):

1. Apparence de cellule: *3D Relaché* (*Format/Feuille de données*)

2. Apparence du texte: *Italique* (*Format/Police*)

Selon le résultat donné à la page suivante (dont l'utilité est plus que douteuse…):

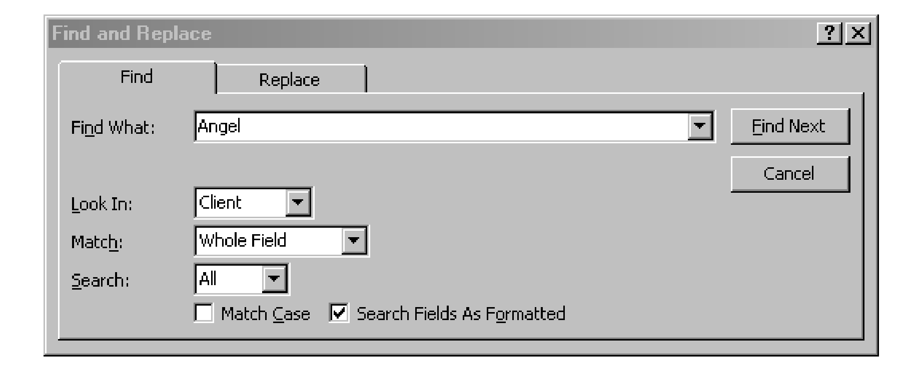
tblArticles : Table

		idArticles	olePhoto	Code Article	Description	Fournisseur	M.O.Q	Prix/unité
	+	*1*		*GEN-001*	*Crayons*	*Dupliburea*	*100*	*SFr. 0.21*
	+	*2*		*GEN-002*	*Enveloppes (10 pces)*	*La maison*	*50*	*SFr. 0.53*
	+	*3*		*GEN-003*	*DIN A4 Papier (500 feuilles)*	*Tartanpion*	*50*	*SFr. 25.04*
▶	+	*4*		*GEN-004*	*Post-It Notes 656*	*Dupliburea*	*60*	*SFr. 10.29*
	+	*9*		*GEN-006*	*Stylos rouges*	*Dupliburea*	*100*	*SFr. 0.53*
	+	*6*		*INF-001*	*Tambour*	*Tartanpion*	*5*	*SFr. 261.45*
	+	*7*		*INF-002*	*Disquette (3.5')*	*Tartanpion*	*1000*	*SFr. 1.37*
	+	*8*		*INF-003*	*Etiquettes LASER (25 feuilles)*	*Tartanpion*	*10*	*SFr. 37.70*
	+	*10*		*INF-004*	*Toner*	*Tartanpion*	*20*	*SFr. 90.20*
*		*(AutoNumber)*						*SFr. 0.00*

4.10 Outil recherche

Le but maintenant est d'apprendre à utiliser l'outil de recherche. Recherchez dans la table *tblSorties* et dans le champ contenant les régions, tous les articles qui ont été vendus au Nord. Répétez l'opération mais à partir d'un formulaire instantané.

Recherchez et remplacez:



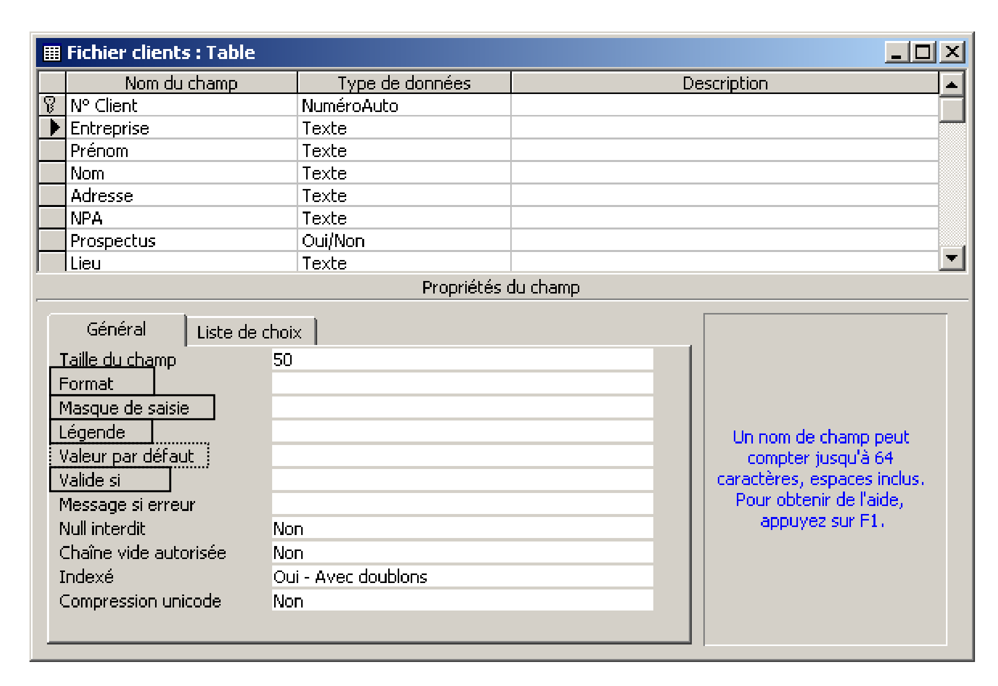
dans toute la table *tblSorties*, le vendeur numéro 2 par le vendeur numéro 11. Refaites la procédure inverse à partir d'un formulaire instantané.

Remarque: L'outil Rechercher/Remplacer gère la wildcard * mais pas les crochets, ni le point d'interrogation. Par ailleurs il ne peut rechercher/remplacer plus de 10'000 données… Le mieux est alors de faire une requête de mise à jour dans lequel le *Critère* de sélection sera la valeur cherchée et la valeur de *Mise-à-jour* le nouveau texte..

Vérifiez l'orthographe dans la colonne *Lieu* de la table *tblFournisseurs*, et ajoutez au besoin les termes dans le dictionnaire personnalisé d'Office.

4.11 Propriétés des tables

Si vous ouvrez une des tables, et que vous la passez en en mode création vous observerez la fenêtre ci-dessous (à quelques détails près).



Remarque: la colonne *Description* est souvent utile au développeur pour prendre des notes quant à l'utilité ou aux spécificités d'un champ. Par défaut, les informations qui y sont saisies seront visibles dans la barre d'état du logiciel lorsque l'utilisateur cliquera sur le champ concerné dans un formulaire. La barre d'état peut cependant être désactivée comme nous le verrons bien plus loin.

Pour chacun des champs de toutes les tables, définissez rigoureusement avec votre formateur les propriétés disponibles pour tous les différents champs.

Testez également ensuite les modifications, soit en mettant à jour votre formulaire existant (le formulaire correspondant à la table) soit en créant un nouveau formulaire automatique (regardez bien le texte en bleu à droite, il permet de comprendre l'utilité des différentes fonctions).

Pour le champ ***Valide Si***, mieux vaut se reporter à l'aide d'Access, les exemples y sont nombreux et c'est un bon exercice pour apprendre à utiliser cette dernière. Cependant il y aura plus tard un exercice ou nous utiliserons l'option *Valide Si*.

Le but principal de l'option **Null Interdit** est d'empêcher l'utilisateur de faire une fausse saisie en écrivant uniquement avec espaces clavier. Evidemment cela peut faire doublon avec le

masque mais le concept de Null Interdit existe dans la grande majorité des SGBR ce qui n'est de loin pas le cas du masque de saisie!

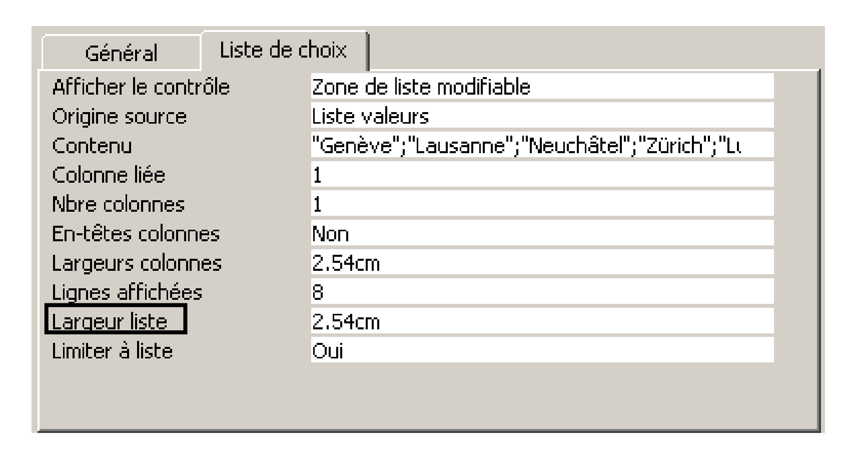
Remarque: **Attention à ne pas mettre des *Null Interdit* partout car sinon vous pourriez ne pas pouvoir créer de données où que ce soit!!! Il s'agirait alors d'une erreur de modélisation de la part du développeur (heureusement c'est une erreur qui prend que quelques minutes à corriger).**

- L'option ***Chaîne vide autorisée*** permet de choisir si on considère l'espace vide comme une valeur d'entrée valable. Normalement on va plutôt dire *Non*…

- La ***Compression Unicode*** doit être activée sur *Oui* si uniquement des caractères latins sont supposés être utilisés dans cette base (les caractères latins pouvant être codés sur 1 octet plutôt que 2).

Depuis Access 2007 on trouve également un champ *Mode IME* et *Mode de formulation IME* (**I**nput **M**ethode **E**ditor) qui concerne les différents caractères japonais Kanji et la manière de les interpréter.

L'onglet appelé *Liste de choix* a également certaines options très intéressantes qu'il faut systématiquement définir (voir capture d'écran page suivante).



Comme vous pouvez le voir ci-dessus, on retrouve les propriétés du champ qui ont été définies lors de l'utilisation de "l'assistant de liste de choix". Cependant l'option "Limiter à la liste" ne nous était pas apparue et elle est pourtant importante.

Vu le nombre de paramètres qu'il faut définir dans Access, il est important de vous rappeler que lorsque vous créez une base, vous devez d'abord commencer par créer les tables et de définir toutes leurs propriétés et contraintes avant même de continuer avec quoi que ce soit!!!

4.12 Tris et filtres

Triez la table *tblArticles* dans l'ordre croissant des *tblDesignation* d'articles.

Filtrez ensuite avec un "filtre par sélection", tous les articles avec le numéro de fournisseur 3 (cliquer dans la cellule avec la valeur 3 et aller dans le menu *Enregistrements/Filtre/Filtrer par sélection*). Désactivez ensuite le filtre en allant dans le menu *Enregistrements/Afficher Tous les enregistrements* ou en cliquant sur l'icône de filtre.

Filtrez tous les articles qui ont un prix supérieur à 50.- (placer le curseur dans la colonne qui doit servir de tri, menu contextuel *Filtrer Pour* et saisissez *>50*)

Il y a plusieurs méthodes pour définir des critères qui utilisent des symboles bien spécifiques à Access. Il est important de bien les comprendre, car ils sont utiles lors de la création de filtres et de requêtes!

Attention!!! Certaines des critères ci-dessous ne fonctionnent que dans des requêtes et pas dans des filtres par sélection.

4.12.1 Critères numériques et textes

> | >= | <> | <= | >=

4.12.2 Critères numériques et dates

K*
Tous les enregistrements dont le contenu du champ spécifié commence par la lettre K

COMME M?ier
LIKE M?ier
Tous les enregistrements dont le contenu du champ spécifié est du type Maier, Moier,…) le ? représente un caractère unique, selon les besoins on écrirait: **M*ier**

#07.07.2002#
Tous les enregistrements dont le contenu du champ spécifié correspond à la date 07.07.2002. Les # ne sont pas obligatoires.

4.12.3 Caractères génériques (wildcards)

Attention! Si quelqu'un a activé l'option de compatibilité SQL ANSI 92 dans les options avancées de MS Access la plupart des commandes ci-dessous ne fonctionneront pas.

***#**
Tous les enregistrements dont le contenu du champ spécifié se termine par un chiffre.

COMME Du[pli]rex
LIKE Du[pli]rex
Tous les enregistrements dont le contenu du champ spécifié contient entre les lettres *Du* et *rex* d'autres caractères, tels que *Duprex*, *Dulrex*, *Duirex*… (vous pouvez choisir le nombre de lettres entre crochets)

COMME Dupon[!t]
LIKE Dupon[!t]

Tout ce qui commence par *Dupon* mais qui ne finit pas avec la lettre *t.*

COMME Du[p-z]
LIKE Du[p-z]
Tous les enregistrements dont le contenu du champ spécifié se termine par *Duprex*, *Durant*..,

la première lettre du terme souligné restant compris entre p et z

>="D"
Tous les enregistrements dont le contenu du champ spécifié commence avec les lettres *D* à *Z*

>3 ET <7
>3 AND <7
Tous les enregistrements dont le contenu des champs est compris entre les deux valeurs spécifiées. Il est donc possible d'utiliser des structures logiques

"A*" OU >"C*"
"A*" OR >"C*"
Tous les enregistrements qui commencent par *A* ou par un caractère plus grand que *C* (attention *Ca*…, *Cb*… *Ch* seront compris dedans étant plus grands que *C*)

<>"G*"
Tous les enregistrements sauf ceux commençant par la lettre *G*

NOT IN ("France";"Allemagne")
PAS ("France";"Allemagne")
Tous les enregistrements qui ne sont ni en *Allemagne* ni en *France*

"K*" OU "L*"
"K*" OR "L*"
Tous les enregistrements commençant soit par la lettre K, soit par la lettre L

4.12.4 Fonctions génériques

Pour exprimer des critères dans l'option *Filtrer Pour*, nous pouvons aussi utiliser des fonctions, mais ces dernières sont spécifiques à la fois du SGBDR et du type de données du champ considéré. Citons quelques exemples classiques:

NbCar([Nom])="4"
Len([Nom])="4"
Retrouve tous les noms de 4 caractères

Droite([Nom];2)="se"
Right([Nom];2)="se"
Retrouve les nom se terminant par "se"

Gauche([Nom];2)="du"
Left([Nom];2)="du"
Retrouve les noms commençant par "du"

ExtracChaine([Nom];2;3)="ach"
Mid([Nom];2;3)="ach"
Retrouve le nom "Machin", lequel contient la chaîne de trois caractères "ach" en commençant au deuxième caractère.

PartDate("aaaa";[datCommande])=2000
DatePart("yyyy";[datCommande])=2000

Retrouve les commandes de l'année 2000. Cette fonction opère aussi (en français) avec "j" pour le jour, "m" pour le mois, et "t" pour l trimestre.

DiffDate("j";[datCommande];[datLivraison])>100
DateDiff("d";[datCommande];[datLivraison])>100
Retrouve les produits qui ont été livrés plus de 100 jours après avoir été commandées. Cette fonction opère aussi (en français) avec "j" pour le jour, "m" pour le mois, et "t" pour l trimestre.

Jour([datCommande])=12
Day([datCommande])=12
Retrouve les commandes effectuées le 12 (des mois présents dans la table). La même écriture est valable avec Mois/Month et Année/Year au lieu de Jour/Day.

4.12.5 Tris et filtres avancés

Maintenant, triez les données de la table *tblVendeurs* dans l'ordre croissant du nom des vendeurs, puis dans l'ordre décroissant des régions de ventes (*Enregistrements/Filtrer/ Filtrage spécial*).

Filtrer de façon à obtenir:

1. Tous les vendeurs de la région Nord

2. Tous les vendeurs qui ne sont pas attribués à la région Nord (attention, MS Access n'est pas "case sensitive")

3. Tous les Vendeurs des régions Nord ou Sud (ligne OU)

4. Tous les vendeurs ayant M ou C pour première lettre du nom et se situant dans la région Nord

   Bien que l'on fasse rarement ainsi, sauvegardez ce filtre comme requête sous le nom *Fichier-Requête* en passant par le menu "Ficher/Enregistrer la requête". On verra plus tard, comment l'on procède traditionnellement pour créer des requêtes.

5. Tous les vendeurs de la région de vente Nord avec un numéro de vendeur supérieur ou égal à 4, triés dans l'ordre croissant du *Nom Vendeur*.

5 Relations (jointures)

La notion de relation est ce qu'il y a de plus important dans MS Access où dans tout SGBDR. C'est ce qui fait de cette famille de logiciel leur toute puissance dans la gestion simple de bases de données plus ou moins complexes. Il est souvent très difficile d'en comprendre le principe, c'est pourquoi nous irons relativement lentement lors de l'étude de ce chapitre.

Remarque: Les relations ont une propriété sous-jacente qui se nomme la "jointure" - techniquement il existe trois types de jointures possibles pour une relation. Une relation a obligatoirement une des 3 jointures, d'où le fait que dans certains ouvrages, les auteurs ne n'utilisent par le terme "relation" car pas assez précis mais uniquement le terme "jointure". Dans le présent le présent chapitre, nous ferons usage du terme "relation" pour signifier qu'implicitement il s'agit de "jointures équivalentes".

Il existe différentes types de relations dont voici la liste (nous en avons déjà fait mention lors de notre introduction au schéma ER page 27):

5.1 Assistant liste de choix

Avant de commencer ce chapitre nous sommes obligés de voir en détails qu'est-ce que l'*Asssistant Liste de Choix* (je ne voulais pas écrire de chapitre à l'origine sur ce sujet trivial que l'on retrouve en masse dans les livres et sur Internet mais suite à de nombreuses demandes j'ai finalement changé d'avis).

Il existe deux formes majeures de l'assistant liste de choix. Nous allons commencer par le plus simple (qui lui-même peut être dérivé en plusieurs sous-catégories) mais qui est aussi le moins recommandé car ne respectant pas la deuxième forme normale.

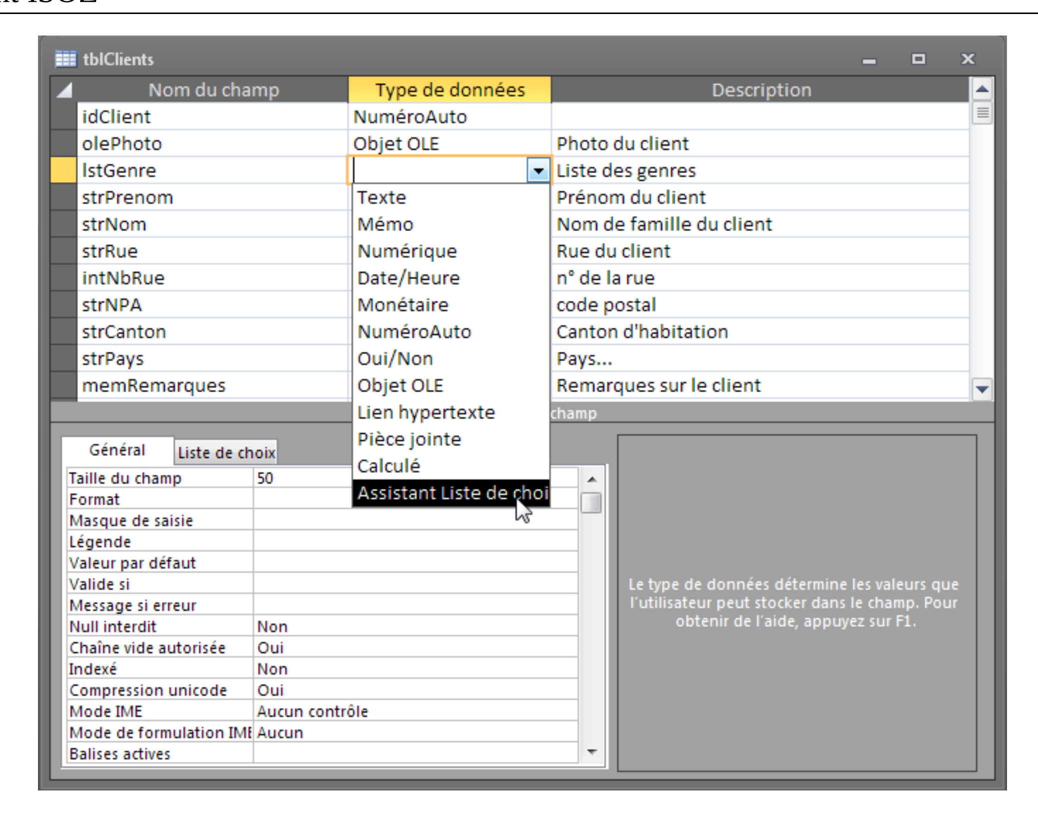
5.1.1 Assistant liste de choix (ALC) statique

Nous allons faire ici un exemple très simple et imagé (avec captures d'écrans) des trois cas de figures les plus courants des listes déroulantes statistiques

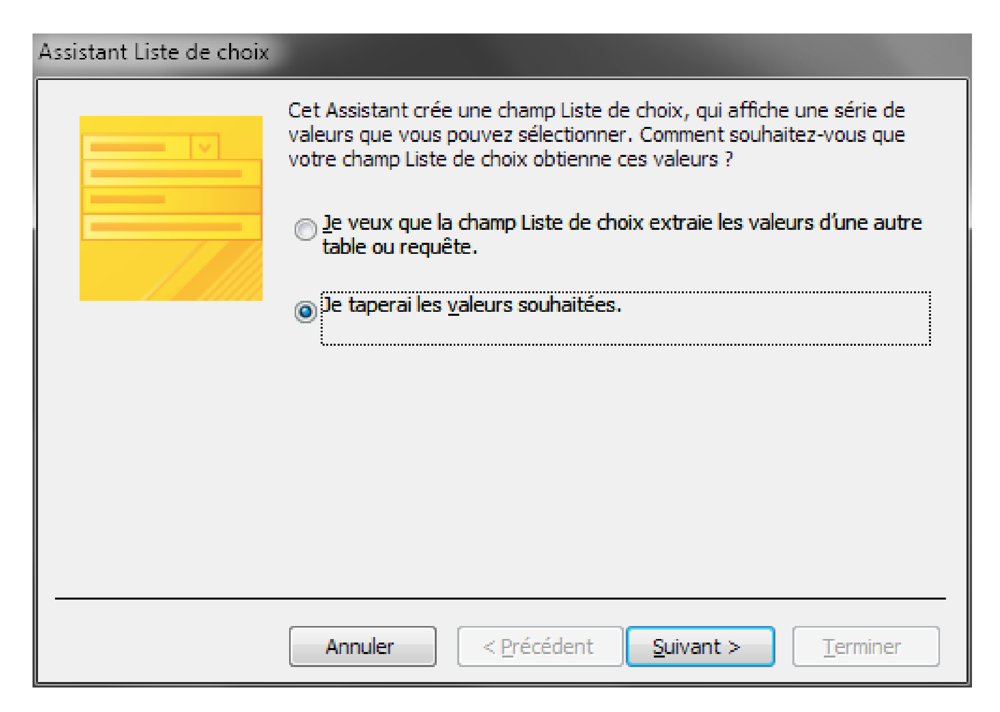
5.1.1.1 ALC statique à choix unique de type zone de liste déroulante

Pour ce cas particulier, nous allons prendre la table *tblClients* pour laquelle nous souhaiterions une liste déroulante statistique à choix unique permettant le choix des genres (M., Mme, Mlle, Dr., etc.).

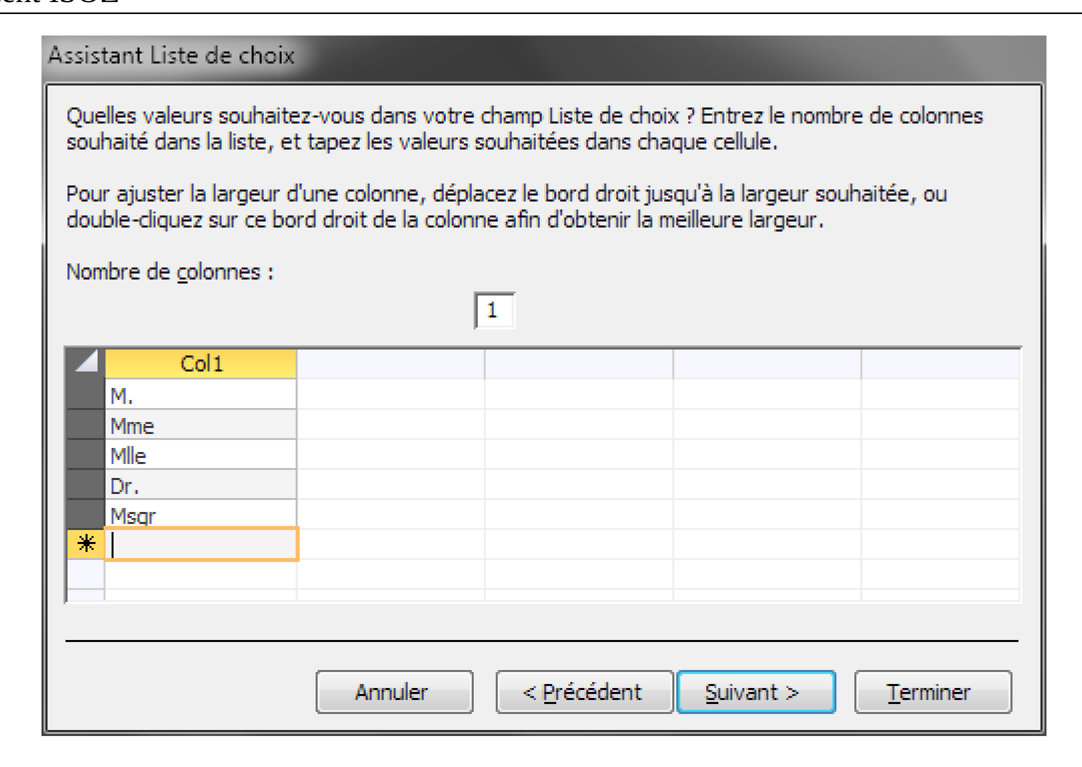
Nous ouvrons donc la table en mode création dans laquelle nous créons un champ nommé *lstGenre* qui sera du type *Texte* en *Assistant liste de choix* (captures effectuées avec MS Access 2010):



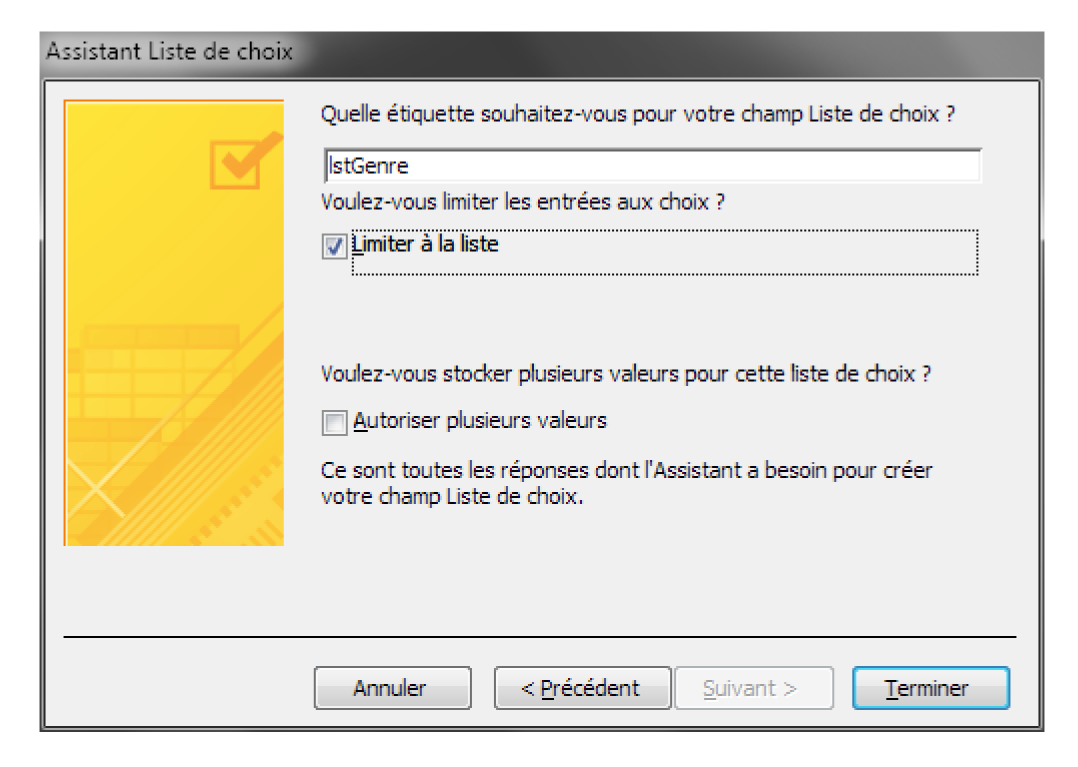
Nous avons alors un assistant qui démarre et à la première étape, nous sélectionnons *Je taperai les valeurs souhaitées*:



et cliquons suir *Suivant* pour y saisir quelques titres par défaut:

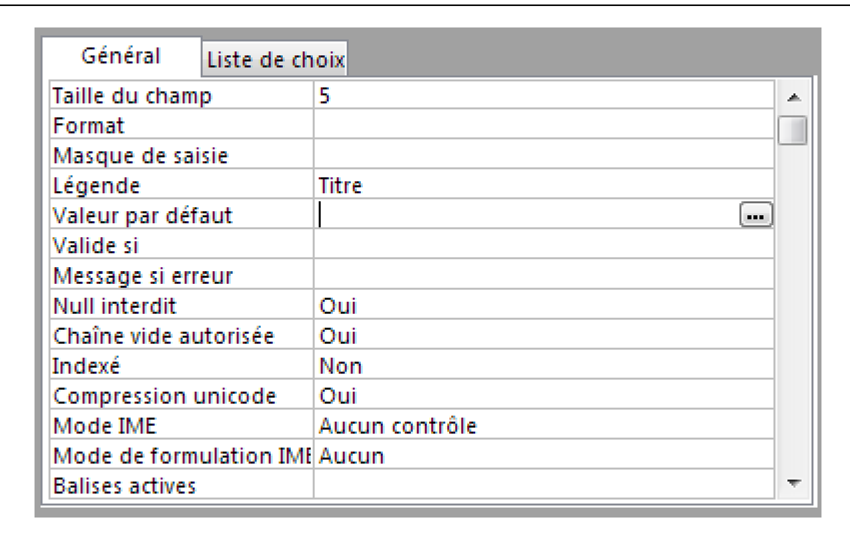


Nous cliquons ensuite sur *Suivant*:

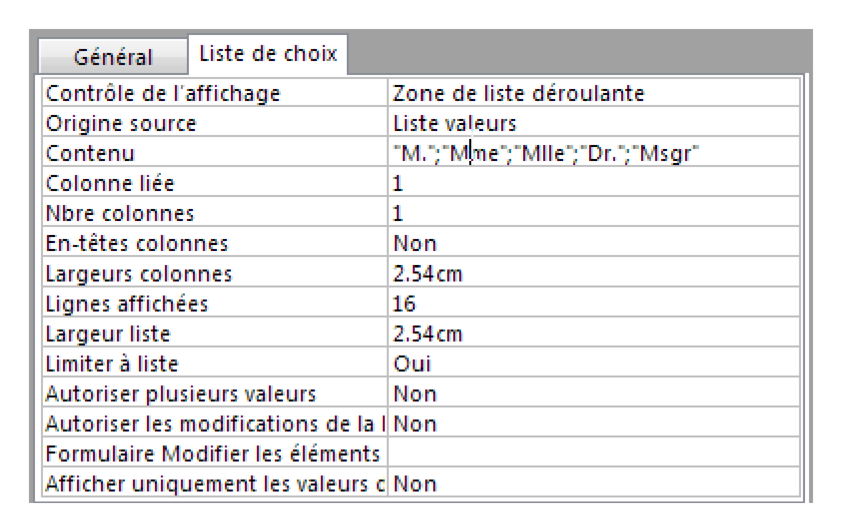


en spécifiant bien que nous ne souhaitons pas que les utilisateurs puissent taper ou choisir autre chose que ce que nous avons défini (*Limiter à la liste*) et qu'il ne peuvent pas sélectionner plusieurs valeurs à la fois (*Autoriser plusieurs valeurs*). Nous cliquons sur *Terminer*.

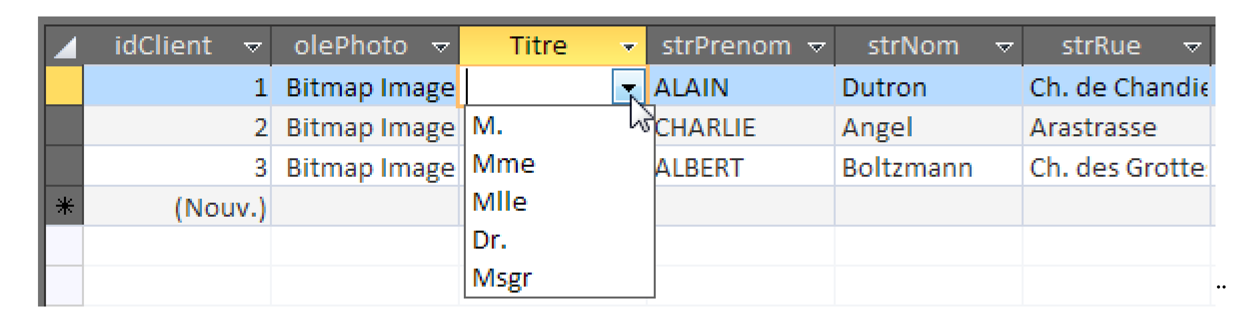
Ensuite, dans les propriétés *Générales* du champs nous allons typiquement prendre:



et nous vérifions bien que dans l'onglet *Liste de choix* nous puissions changer éventuellement plus tard les paramètres:



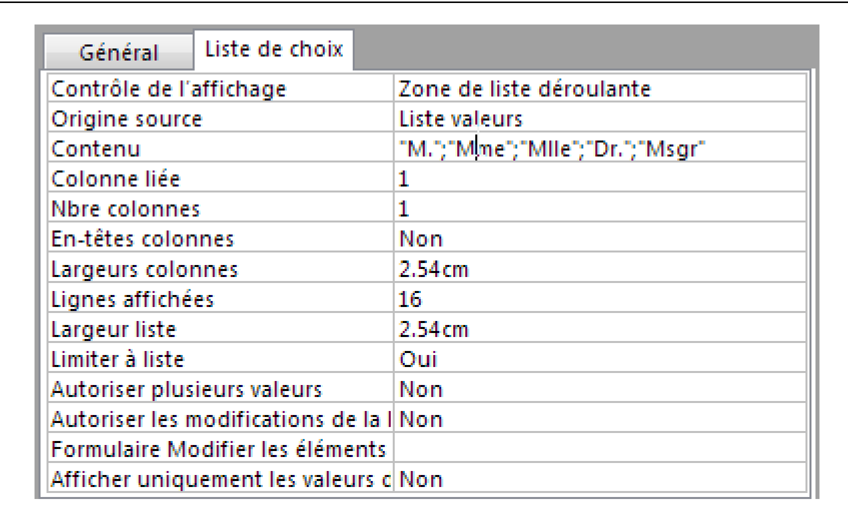
et la liste déroulante statistique à choix unique est alors directement utilisable en mode table:



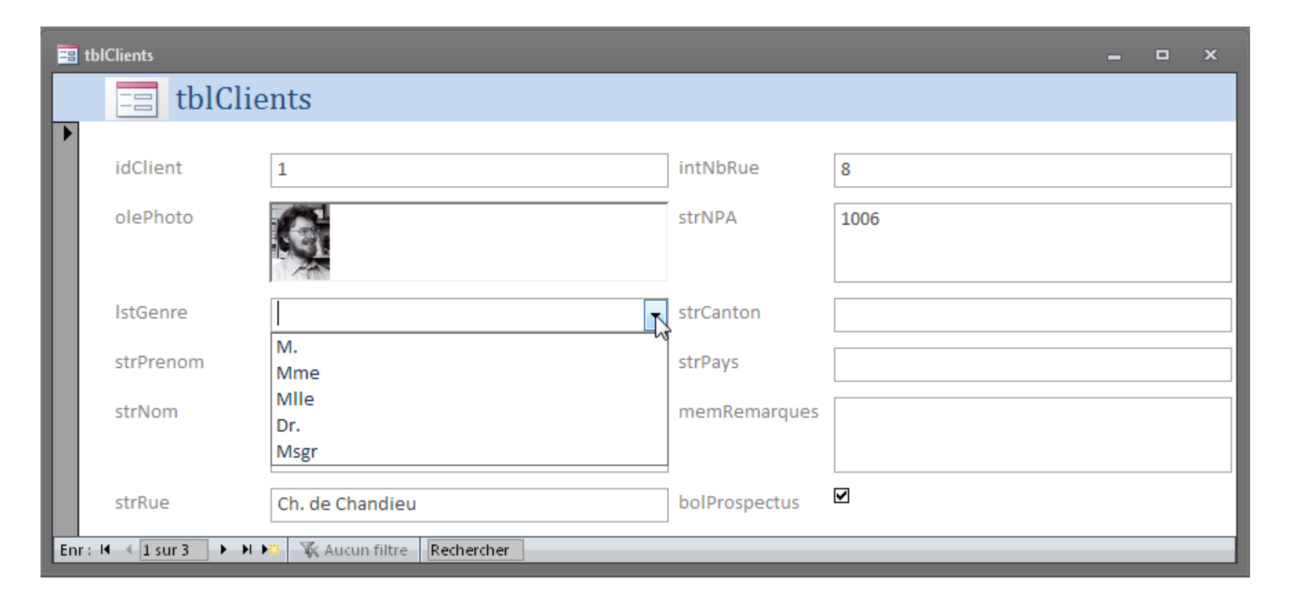
Remarque: Ceci dit, n'importe quel connaisseur vous dira qu'il vaut mieux faire une liste à choix liée à une table avec une intégrité référentielle que d'utiliser cette technique!

5.1.1.2 ALC statique à choix unique de type zone de liste

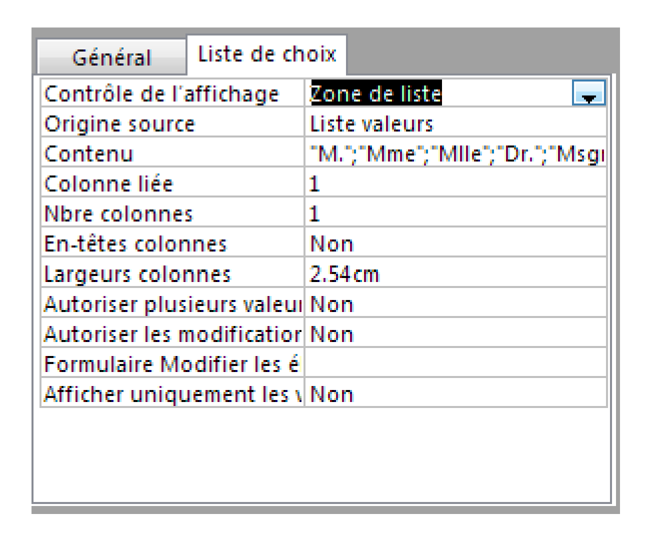
La procédure est en tout point identique à la précédente. La seule chose qui change se situe au niveau de l'option *Contrôle de l'affichage* ci-dessous:



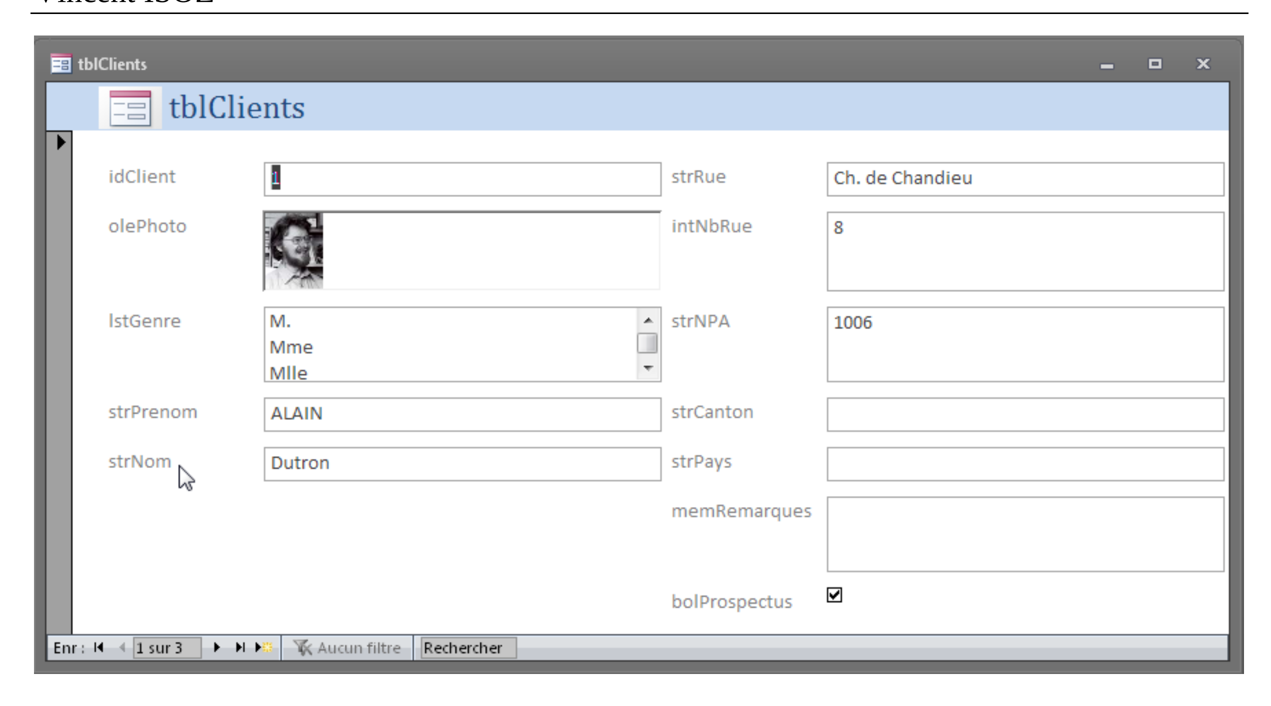
Effectivement, dans l'exemple précédent l'option a pour valeur (comme le montre la capture ci-dessous) *Zone de liste déroulante*. Ce qui donne dans un formulaire le résultat suivant:



Mais si nous changeons la même option avec la valeur *Zone de liste*:



Alors nous obtenons:



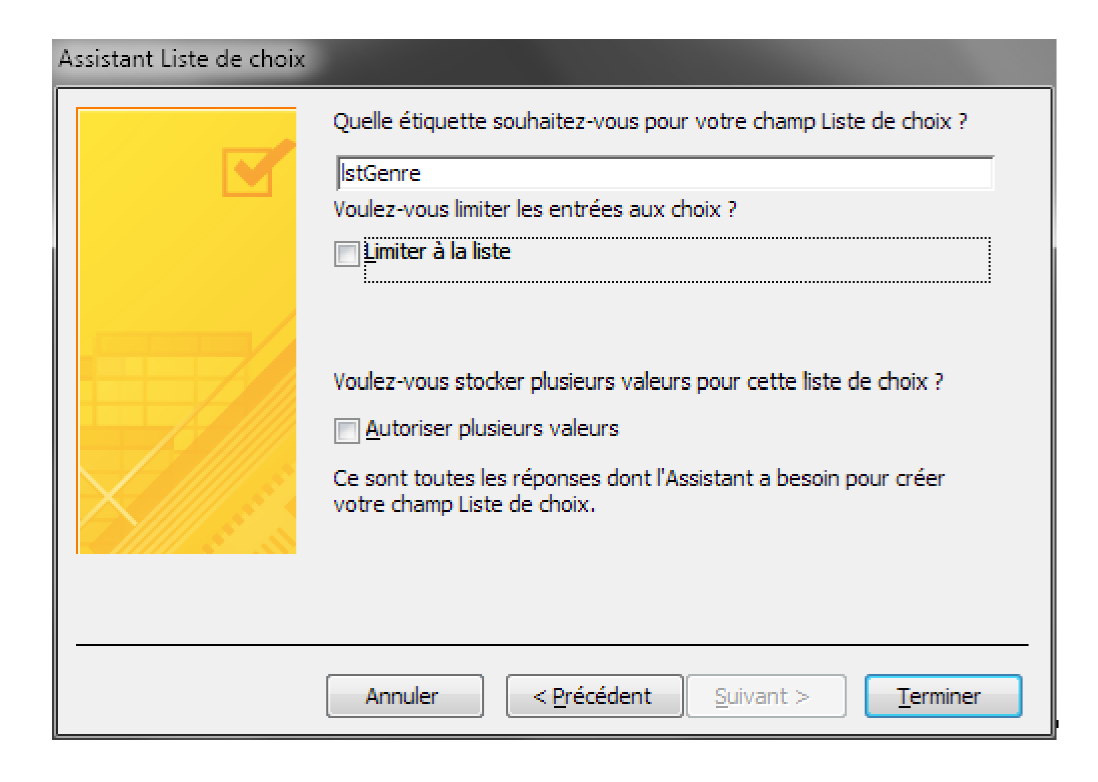
et là nous voyons bien la différence entre une zone de liste déroulante et une zone de liste simple. Mais la différence n'est visible qu'avec les formulaires.

Remarque: Encore une fois, n'importe quel connaisseur vous dira qu'il vaut mieux faire une liste à choix liée à une table avec une intégrité référentielle que d'utiliser cette technique!

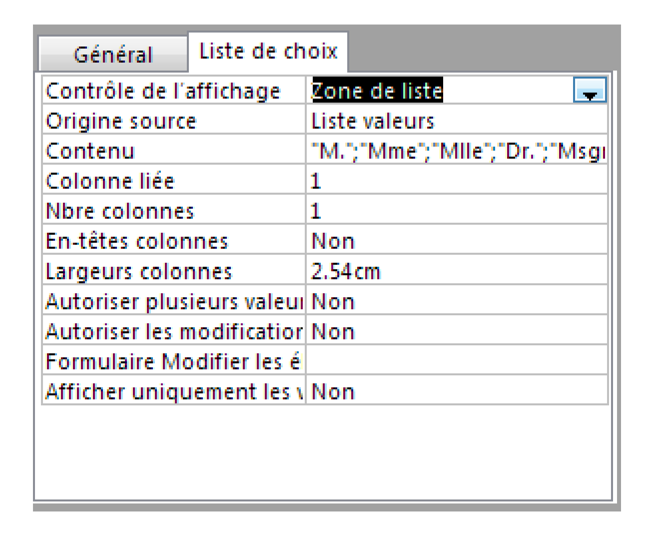
5.1.1.3 ALC statique à choix unique de type liste déroulante extensible

Attention! Ce type de liste ne fonctionne pas avec le masque de saisie!

La méthode est la même que l'antéprécédente mais à la différence que cette fois-ci, nous décochons l'option *Limiter à la liste*:

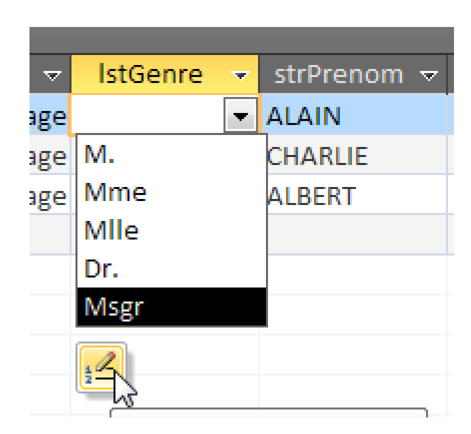


option qui peut aussi être changée à tout moment des les options avancées des listes:



En mettant la propriété *Autoriser les modifications* à la valeur *Oui*.

La saisie alors comporte, que ce soit dans un formulaire ou dans une table, une option complémentaire dans la liste comme le montre bien la capture ci-dessous:



et si nous cliquons sur ce petit bouton vient alors:



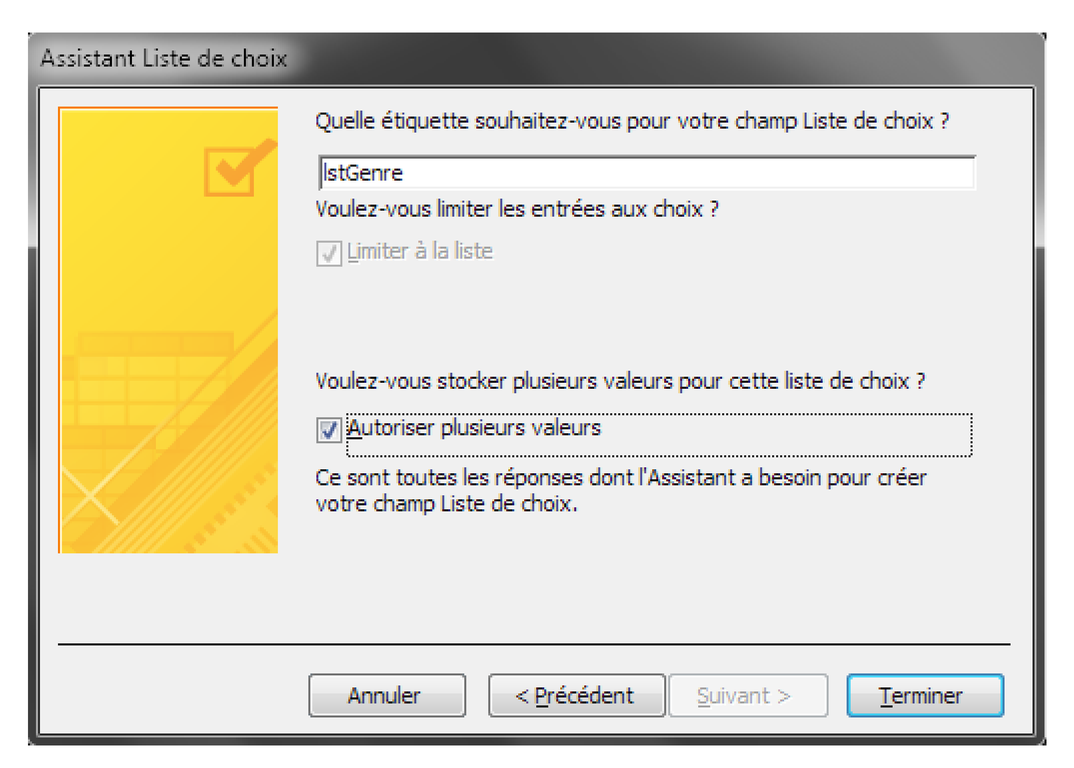
La suppression d'une des valeurs de la liste ne la supprimera pas dans les enregistrements où elle est déjà utilisée.

Malheureusement ce type de liste ne sera pas trié dans l'ordre alphabétique!

Remarque: Encore une fois, n'importe quel connaisseur vous dira qu'il vaut mieux faire une liste à choix liée à une table avec une intégrité référentielle que d'utiliser cette technique!

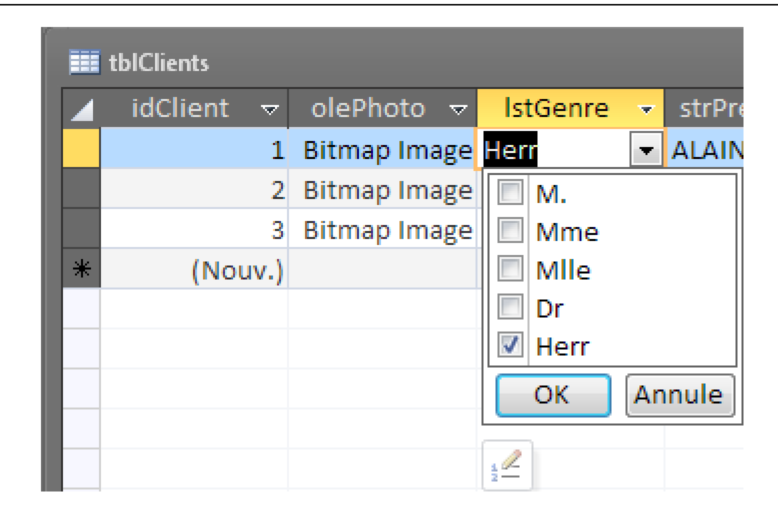
5.1.1.4 ALC statique à choix multiple de type liste déroulante extensible

La méthode est la même que la précédente mais à la différence que cette fois-ci, nous cochons l'option *Autoriser plusieurs valeurs*:

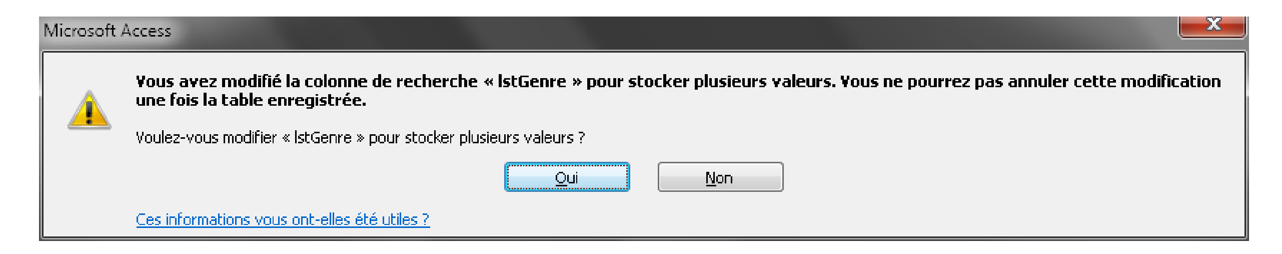


Vous remarquerez dès lors que l'option *Limiter à la liste* se cochera même si vous l'avez décochée au préalable!

Nous avons alors (ce qui a peu de sens dans le cas présent), la possibilité de choisir plusieurs genres pour un même client:



Lorsque vous faites ce choix, vous avez très probablement remarqué le message suivant:



Et effectivement, vous ne pourrez plus changer le type de ce champ. Il vous faudra dès lors le supprimer entièrement.

Remarque: Encore une fois, n'importe quel connaisseur vous dira qu'il vaut mieux faire une liste à choix multiple liée à une table avec une intégrité référentielle que d'utiliser cette technique!

Concernant l'option *Afficher uniquement les valeurs de la source* visible ci-dessous:

Contrôle de l'affichage	Zone de liste déroulante
Origine source	Liste valeurs
Contenu	"M.";"Mlle";"Mme"
Colonne liée	1
Nbre colonnes	1
En-têtes colonnes	Non
Largeurs colonnes	2.54cm
Lignes affichées	16
Largeur liste	2.54cm
Limiter à liste	Non
Autoriser plusieurs valeurs	Oui
Autoriser les modifications de la liste de valeurs	Oui
Formulaire Modifier les éléments de liste	
Afficher uniquement les valeurs de la source	Oui

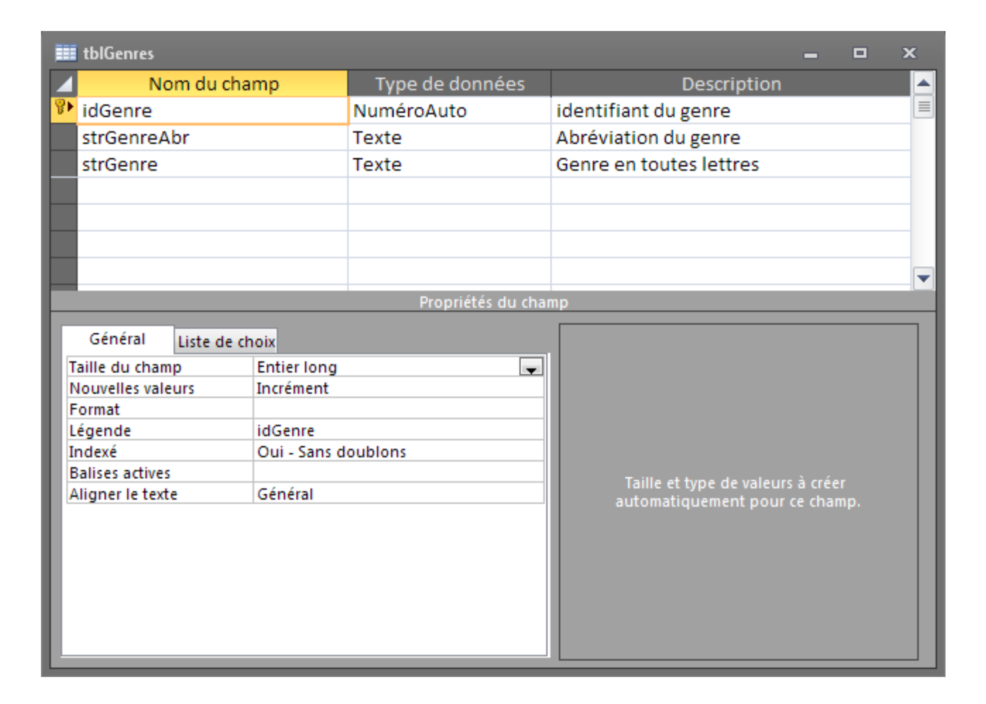
D'abord, elle ne marche que si et seulement si l'option *Autoriser plusieurs valeurs* et sur *Oui*.

Par défaut elle est sur *Non* ce qui signifie que dans la table *tblClients* elle laissera visible dans les cellules de colonne *lstGenre* les valeurs qui ne se trouvent pas dans la liste déroulante à choix multiple. Si vous la passez à *Oui* elle masquera (et non supprimera!) de la colonne *lstGenre* toutes les valeurs non disponibles dans la liste déroulante.

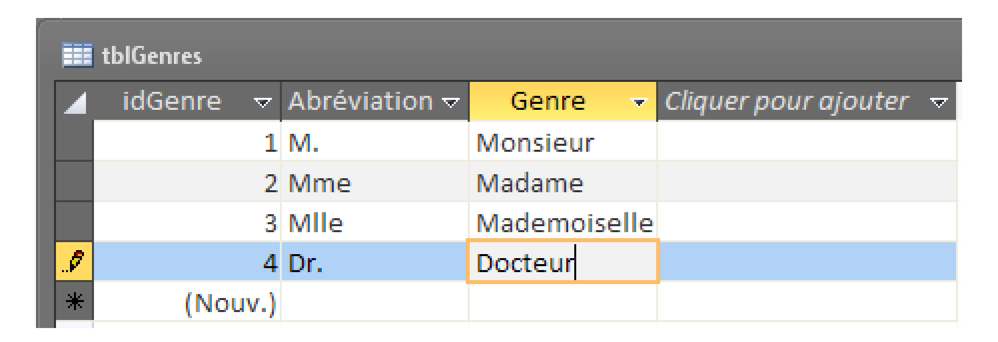
5.1.2 Assistant liste de choix (ALC) lié

Les listes de choix vues jusqu'à maintenant posent d'énormes problèmes en matière de mémoire, de rapidité et de migration des informations. Dans tous les cas, il faut lui préférer la même technique mais associant une table externe.

Pour faire les exemples qui vont suivre, nous aurons au préalable préparés une table *tblGenres*:



avec les quelques données suivantes:



5.1.2.1 ALC statique lié à choix unique de type zone de liste déroulante

Pour ce cas particulier, nous allons prendre la table *tblClients* pour laquelle nous souhaiterions une liste déroulante statistique à choix unique permettant le choix des genres (M., Mme, Mlle, Dr., etc.).

Nous ouvrons donc la table en mode création dans laquelle nous créons un champ nommé *lstGenre* qui sera du type *Texte* en *Assistant liste de choix* (captures effectuées avec MS Access 2010):

tblClients

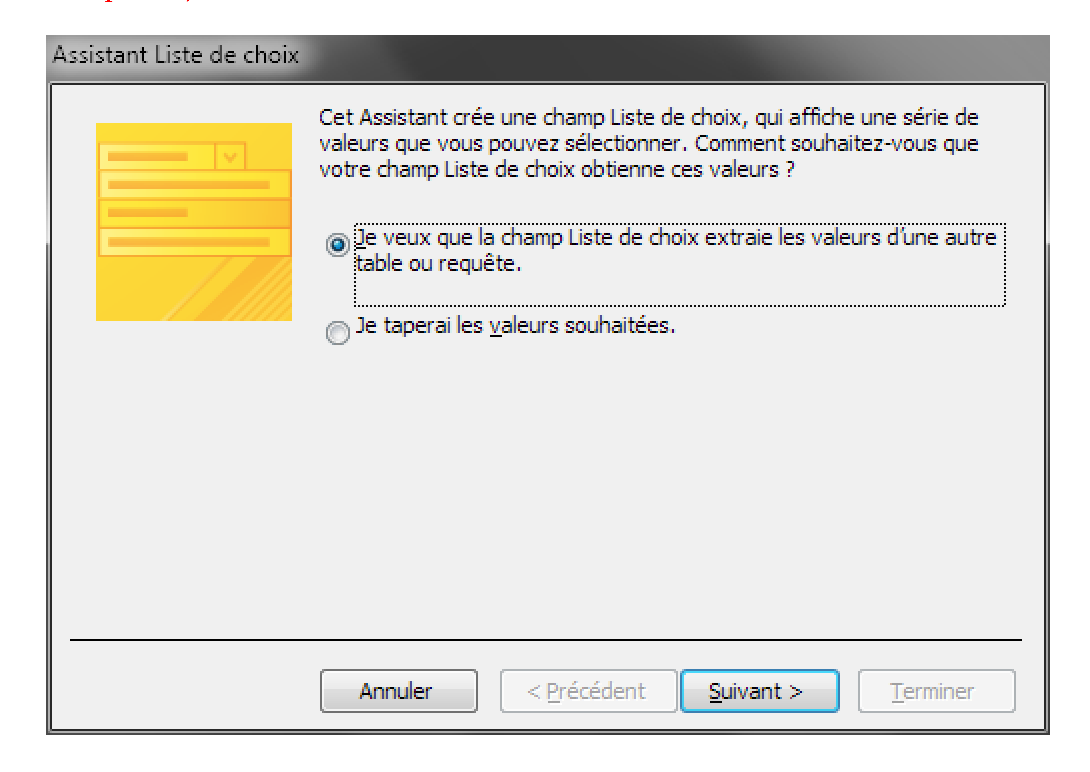
Nom du champ	Type de données	Description
idClient	NuméroAuto	
olePhoto	Objet OLE	Photo du client
lstGenre		Liste des genres
strPrenom	Texte	Prénom du client
strNom	Mémo	Nom de famille du client
strRue	Numérique	Rue du client
intNbRue	Date/Heure	n° de la rue
strNPA	Monétaire	code postal
strCanton	NuméroAuto	Canton d'habitation
strPays	Oui/Non	Pays...
memRemarques	Objet OLE	Remarques sur le client
	Lien hypertexte	
	Pièce jointe	
	Calculé	
	Assistant Liste de choi	

Général | Liste de choix

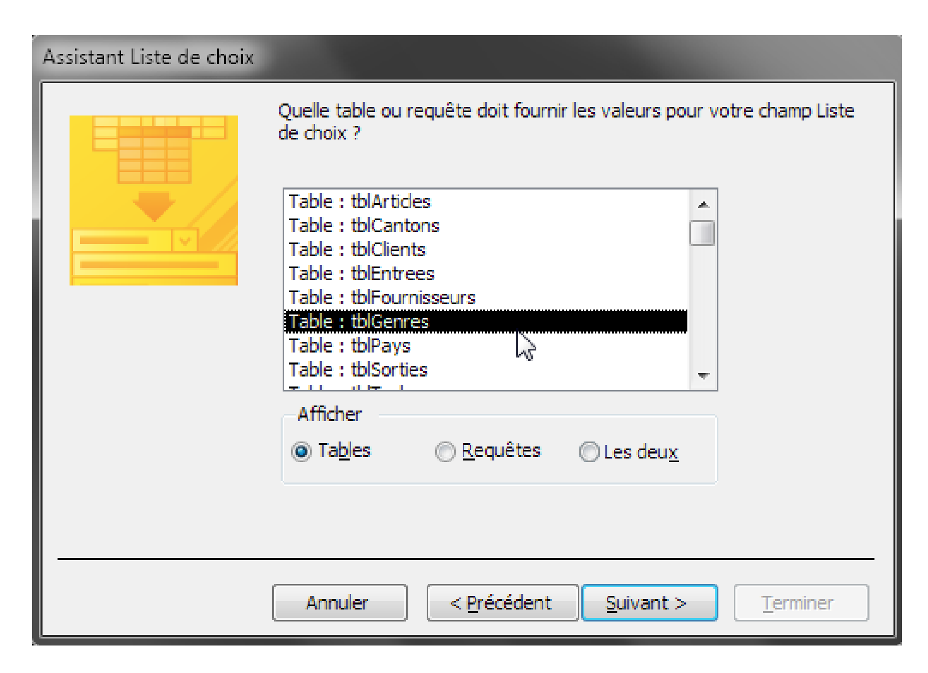
Taille du champ	50
Format	
Masque de saisie	
Légende	
Valeur par défaut	
Valide si	
Message si erreur	
Null interdit	Non
Chaîne vide autorisée	Oui
Indexé	Non
Compression unicode	Oui
Mode IME	Aucun contrôle
Mode de formulation IME	Aucun
Balises actives	

Le type de données détermine les valeurs que l'utilisateur peut stocker dans le champ. Pour obtenir de l'aide, appuyez sur F1.

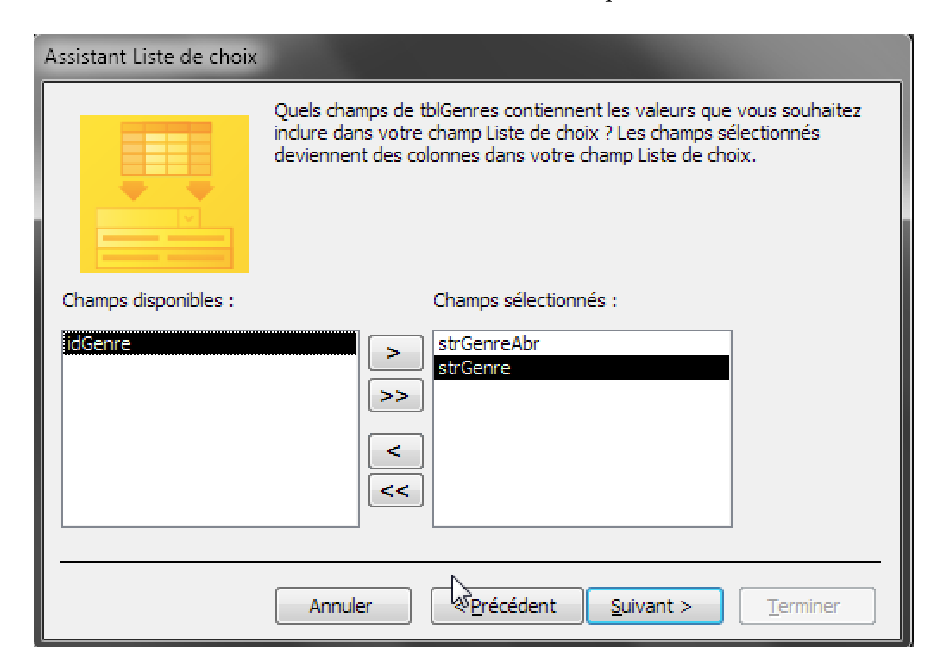
Nous avons alors un assistant qui démarre et à la première étape, nous sélectionnons *Je que que la champs Liste de choix extraie les valeurs d'une autre table ou requête* (vous remarquerez que j'ai même recopié la faute d'orthographe de l'entreprise à 50 milliards de dollars de capital...):



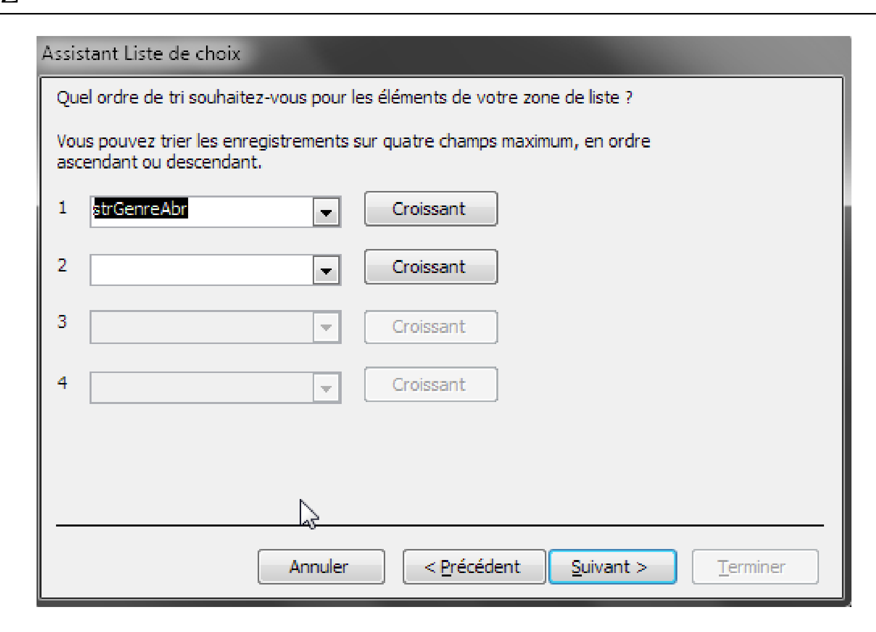
Nous cliquons sur *Suivant*:



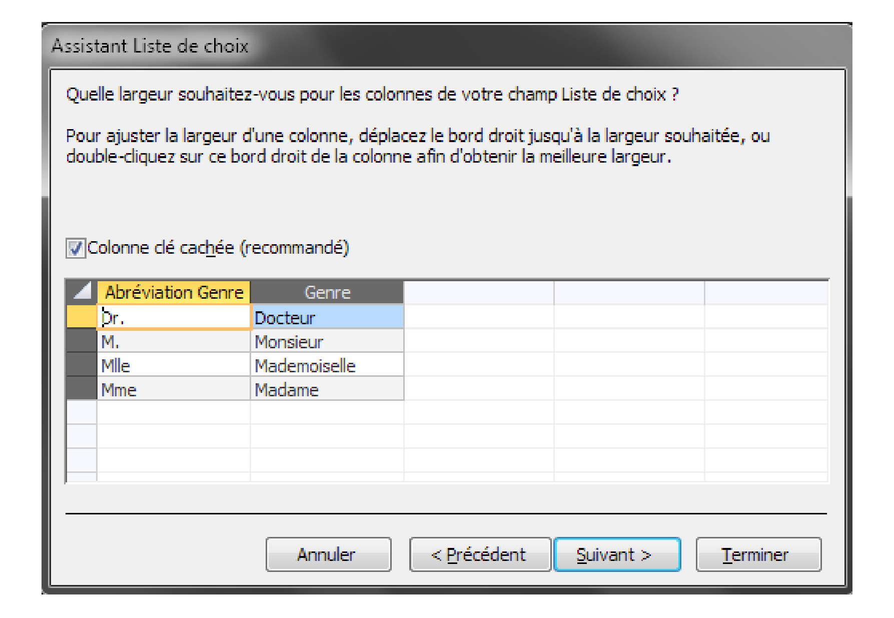
Nous nous limiterons aux *Tables* étant donnée que nous n'avons pas encore étudié les requêtes mais cela n'enlève à rien l'intérêt (potentiel) très puissant qu'il a y de lier une liste à choix à une requête!!!! Nous choisissons la table *tblGenres* et cliquons sur *Suivant*:



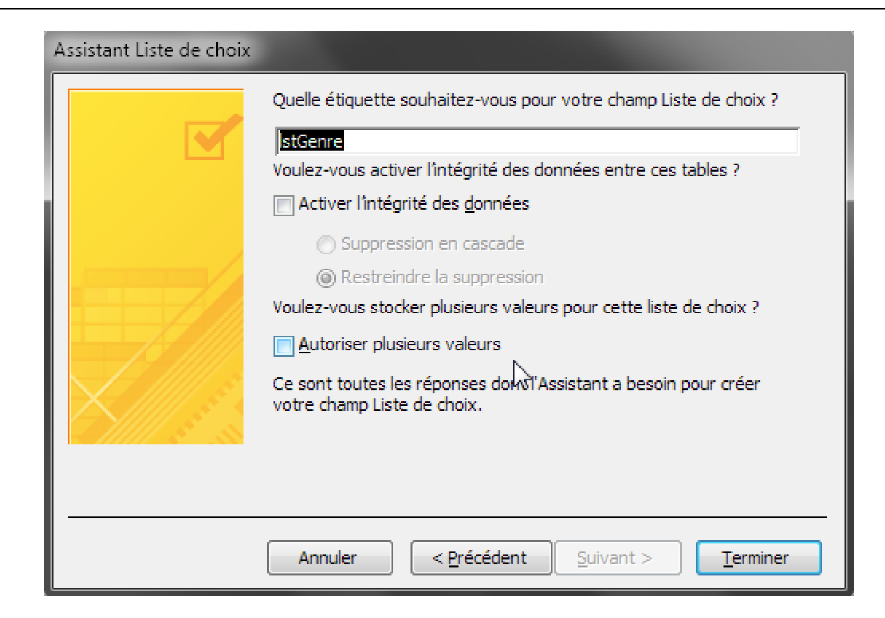
on prend les deux champs (pour les utilisateurs qui ne comprendraient pas les abréviations...) et nous cliquons sur *Suivant*:



Nous trions sur la base des abréviations seulement et nous cliquons sur *Suivant*:



C'est satisfaisant. Nous cliquons sur *Suivant*:

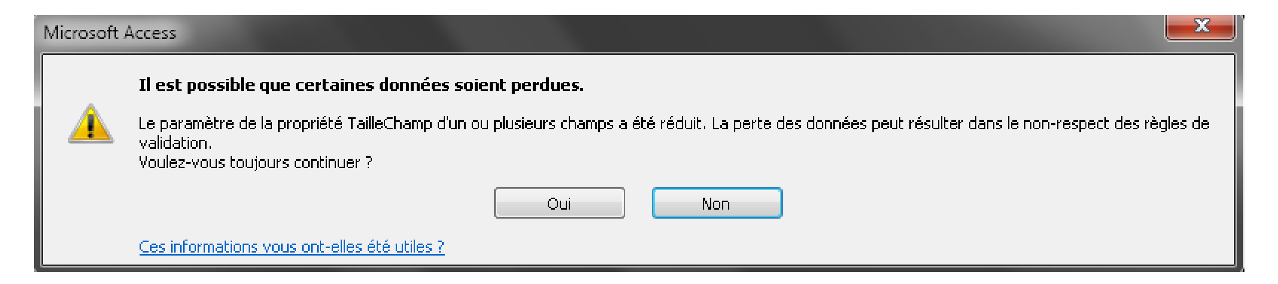


Nous n'activons pas l'intégrité (nous ne voulons pas que les utilisateurs puissent supprimes des Genres) et nous n'autorisons pas la sélection de plusieurs genres.

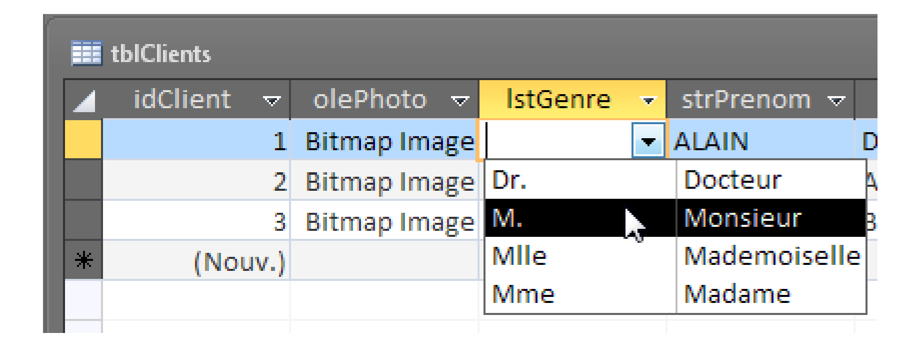
Nous cliquons sur *Terminer*:



Nous validons par *Oui*:



Comme le champs *lstGenres* était initialement un champ *Texte* et qu'il a besoin de le transformer en numérique (Entier Long) pour le lier à la clé primaire de la table *tblGenres* nous validons par *OK*:



et nous avons le résultat souhaité.

Nous pourrions refaire tous les exemples que nous avions présentés avec les listes statiques mais ce serait du gaspillage de page...

5.2 Relation un à un

Exemple 1: Une table Étudiants ⇔ table Taille, autrement dit: un étudiant à une seule taille (à un moment donné de sa vie…)

Exemple 2: une table classes ⇔ table responsables, autrement dit: une classe a un responsable et un responsable n'a la charge que d'une classe

Les deux exemples ci-dessus sont ceux que les spécialistes de l'algèbre relationnelle appellent une "dépendance fonctionnelle".

Prenons un exemple pratique avec notre base de données. Un vendeur peut travailler dans plusieurs succursales au cours de sa carrière (c'est normal) mais nous allons supposer qu'un vendeur ne peut être responsable que d'un succursale tout au long de sa carrière.

Prenons la table *tblVendeurs* de notre base de données *bddMagasin*:

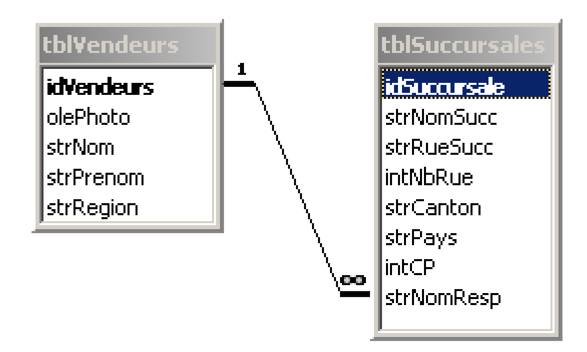


et créons une table de succursales:

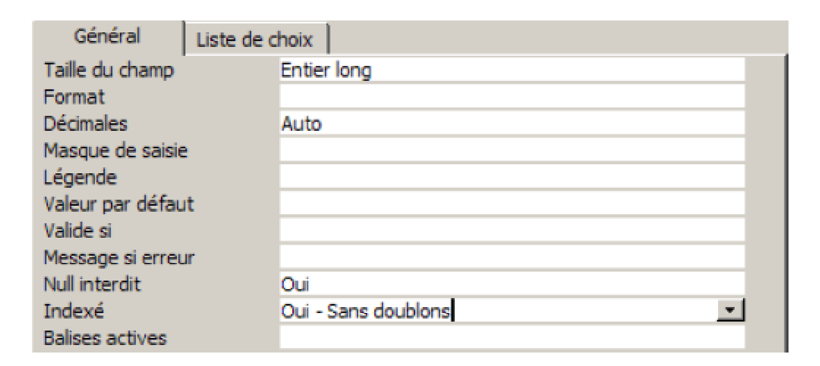
tblSuccursales : Table

	Field Name	Data Type
🔑	idSuccursale	AutoNumber
	strNomSucc	Text
	strRueSucc	Text
	intNbRue	Number
	strCanton	Text
	strPays	Text
	intCP	Number
▶	strNomResp	Number

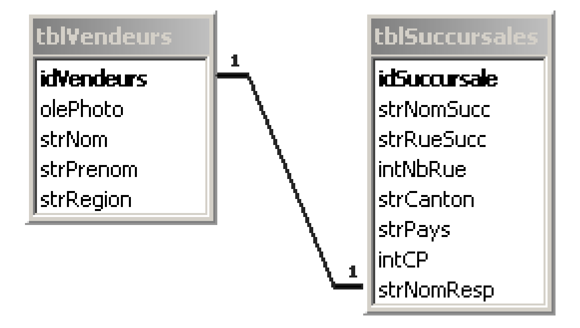
Attention, dans cet exemple il est particulièrement important de ne pas oublier de faire (avec l'assistant liste de choix pour simplifier) une relation entre le champ *strNomResp* et le champ *strNom* de la table *tblVendeur* et ensuite d'aller activer l'intégrité référentielle entre les deux tables:



Allez ensuite dans le mode création de la table *tblSuccursales* et changez l'index ET le Null interdit comme indiqué ci-dessous du champ *strNomResp*:



Si vous revenez ensuite dans le schéma de votre base vous verrez alors que nous avons bien maintenant une relation 1 à 1:



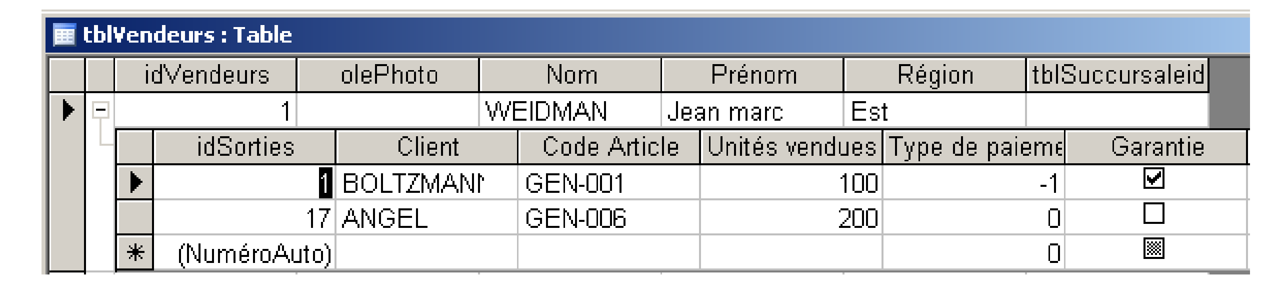
5.3 Relation un à plusieurs

Exemple: une table étudiants ⇔ table livres, autrement dit un étudiant emprunte plusieurs livres et un livre ne peut être emprunté que par un étudiant.

Remarque: La technique de travail avec les relations un à plusieurs nécessite obligatoirement la création d'un champ dans la table "plusieurs" appelé "clé étrangère" (fk). Ce champ contiendra des informations provenant du champ "clé primaire" (pk/id) de la table "un".

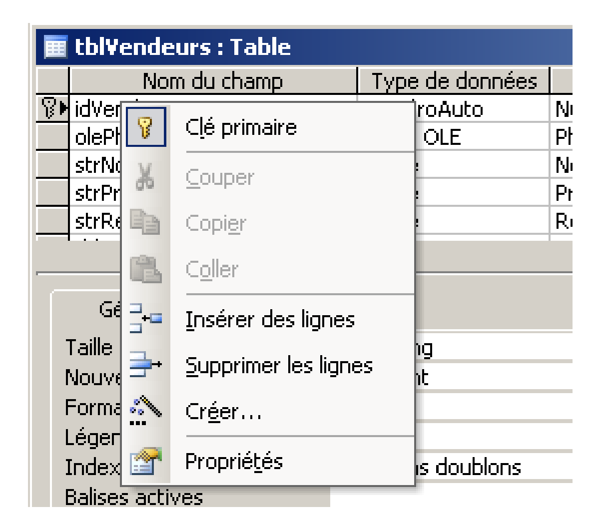
La clé primaire se champ toujours sur un champ à identifiant unique c'est la raison pour laquelle nous choisissons le plus souvent dans le domaine de l'informatique de mettre ces clés sur des champs ID en Auto-Number (cela permet aussi de gagner de la place mémoire en l'occurrence).

Si vous ouvrez la table maître d'une relation un à plusieurs vous verrez une colonne avec un petit + sur la gauche comme ci-dessous:

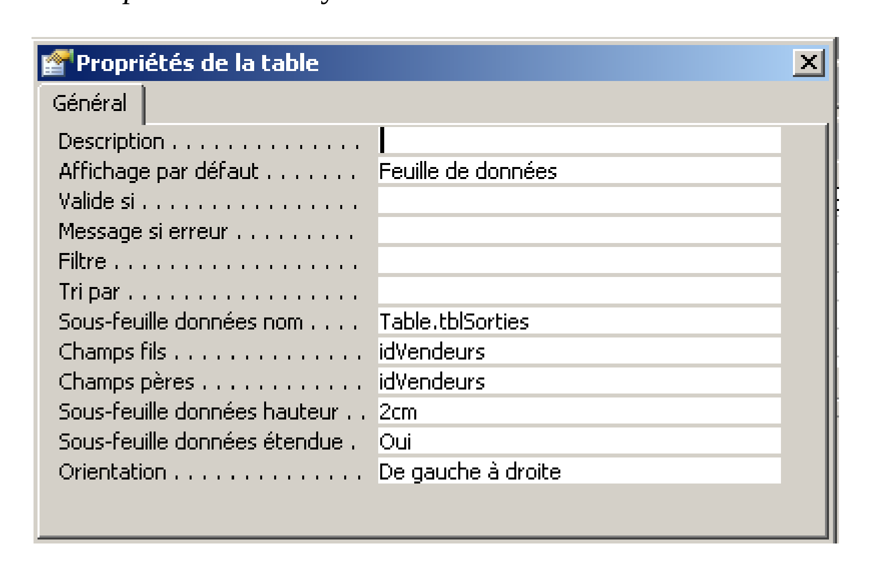


Si la table *tblVendeurs* n'est liée qu'à une seule table par une relation un à plusieurs alors MS Access vous imbrique immédiatement la table ou "sous-feuille" esclave après un clic sur le petit +.

En faisant un clic droit dans le mode création de la table *tblVendeurs*:



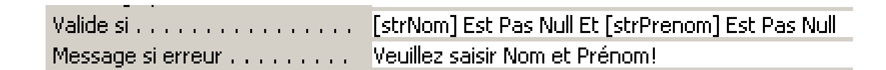
et en sélectionnant *Propriétés* nous voyons cela:



Donnons une description de chaque champ:

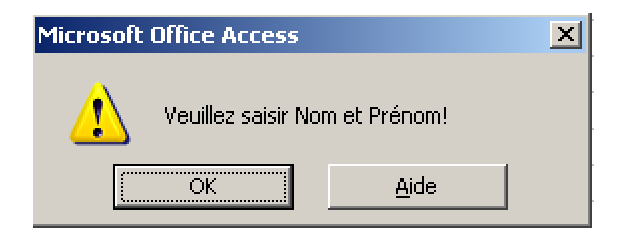
- *Affichage par défaut*: La manière dont la table doit s'afficher lorsque nous l'ouvrons.

- *Valide si*: Il est possible ici de mettre des critères pour valider un enregistrement en son entier. Un bon exemple dans la table *tblVendeurs* serait:



Remarque: Il est également autorisé d'y mettre des formules du type [NomDuChamp]=[NomChamp1]+[NomChamp2]. Ainsi, tant que [NomDuChamp] ne sera pas égal à la somme, l'enregistrement ne sera pas validé!

Ce qui obligerait un utilisateur à saisir le Nom et le Prénom d'un vendeur avant de pouvoir passer à une autre saisie. Et si ceci n'est pas respecté apparaîtrait alors le message d'erreur indiqué:

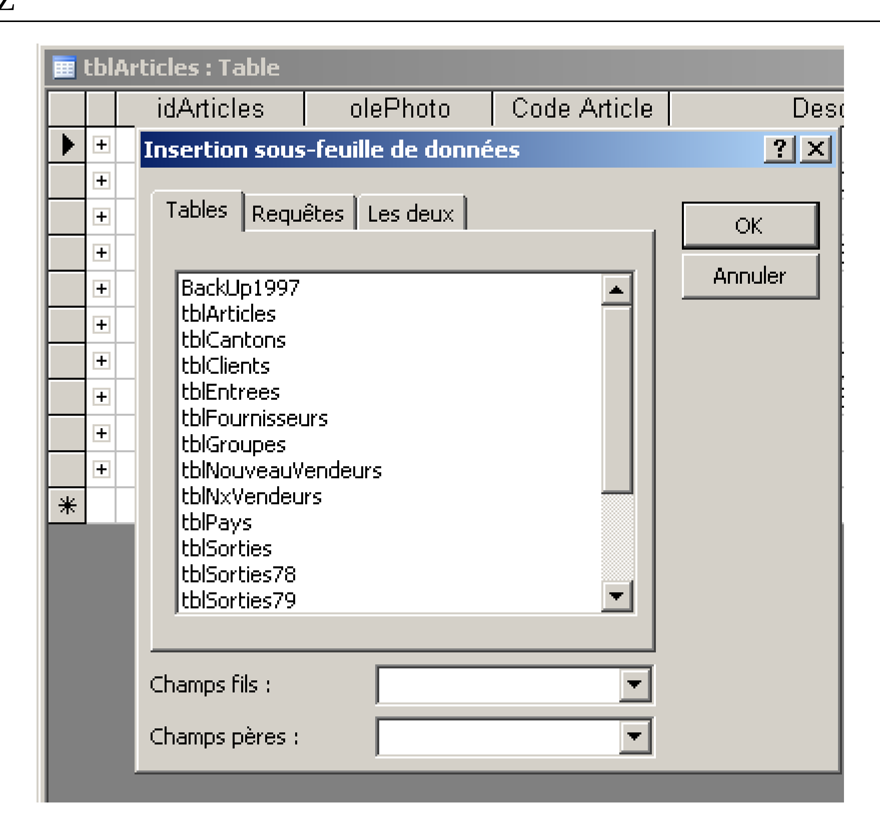


- *Filtre* et *Tri*: Permettent qu'un filtre et un tri soient appliqués automatiquement à chaque fois que la table est ouverte. C'est rarement utilisé même si cela marche très bien.

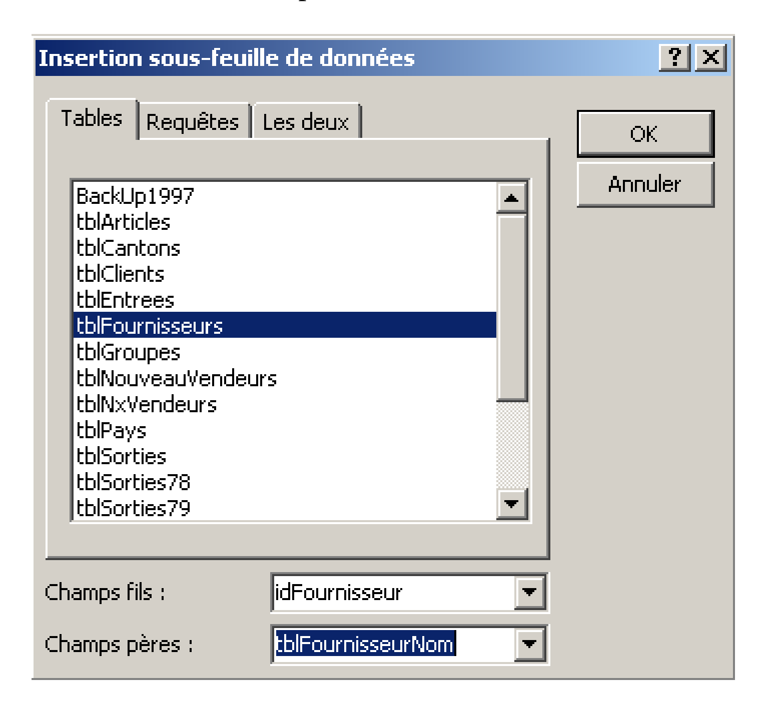
- *Sous-feuille de données nom*: indique la sous-feuille à ouvrir par défaut si la table en question est reliée avec plusieurs tables par une relation un à plusieurs.

- *Champ fils* et *Champ père*: Sont respectivement la clé étrangère et primaire de la relation définie par la table sur laquelle vous êtes et celle choisie dans le champ *Sous-feuilles de données*.

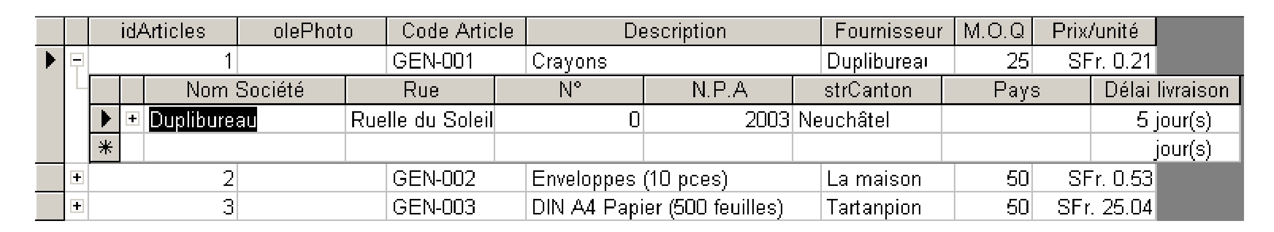
Si vous ouvrez une table MS Access qui est liée à plusieurs tables en un à plusieurs, alors le logiciel ne sachant pas quoi faire va vous demander quelle liaison avec quelle table vous souhaitez visualiser. Par exemple avec la table *tblArticles*, en cliquant sur le petit + nous obtenons:



Il faut ensuite faire un choix un tant soit peu censé:



et valider par OK:



Ainsi, en définissant directement dans les propriétés de la table, la sous-feuille de données, le champ père et le champ fils il ne sera plus utile de faire cette manipulation à chaque ouverture de la table.

- *Sous-feuille de données hauteur*: c'est simplement la hauteur fixe en centimètres que doit avoir par défaut chaque sous-feuille dans la table mère. C'est très pratique si vous avez à chaque fois une sous-feuille de données avec 5000 lignes cela permet de restreindre à seulement quelques unes.

- *Sous-feuille de données étendue*: Permet simplement que lorsque la table est ouverte d'avoir toutes les sous-feuilles de données ouvertes (comme si pour chaque ligne nous avions déjà cliqué sur le petit +).

5.4 Relation plusieurs à plusieurs

Exemple: une table Étudiants ⇔ une table Cours, autrement dit: un étudiant suit plusieurs cours, un cours est suivi par plusieurs étudiants

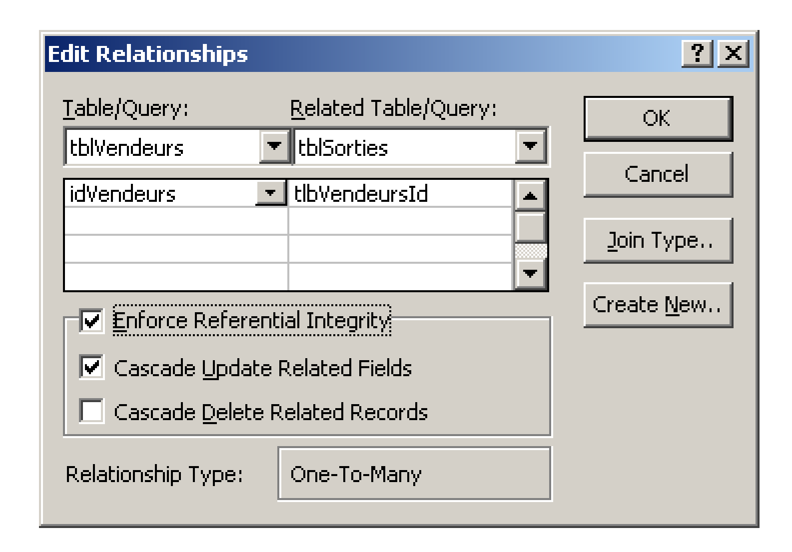
Remarque: la technique de travail avec les relations plusieurs à plusieurs nécessite obligatoirement la création d'une table intermédiaire dite "table de jonction/transition" qui se crée en fait très naturellement.

5.5 Relation "simple" (sans intégrité référentielle)

Vous ne pouvez effacer ou modifier un enregistrement qui est lié au contenu d'une autre table. De plus, la table contenant la clé étrangère peut contenir des éléments ne correspondant à aucun élément de la table contenant la clé primaire. C'est assez dangereux mais le choix dépend des besoins de l'entreprise (veut-elle assurer "l'intégrité" de ses données ou non ?).

5.6 Relation avec intégrité référentielle

En gros (fonction principale mais on reviendra plus en détail là-dessus) par un double clic sur une relation dans la vue des relations vous aurez la boîte de dialogue suivante:



1. *Activer l'intégrité référentielle* (Enforce Referentiel Integrity):

Si vous effacez un enregistrement, ceux qui y font référence seront bloqués mais le choix est quand même possible (sous-options). Vous n'avez aussi plus la possibilité d'avoir dans la table contenant la clé étrangère de la relation, d'éléments qui ne correspondent pas au contenu de la table contenant la primaire de la relation (d'où le terme "intégrité").

2. *Mise à jour en cascade*:

Cette option est disponible seulement si vous avez activé l'intégrité référentielle. Elle est utile aux personnes qui auraient fait l'erreur de mettre leur clé primaire ailleurs que sur un champ ID (numéro auto). Nous n'en ferons donc pas usage dans ce support.

Cette option est par contre redondante (ce qui ne signifie pas qu'il ne faut quand même pas l'activer) si vos clés étrangères sont basées sur des listes à choix.

3. *Suppression en cascade*:

A pour fonction de supprimer tous les enregistrements reliés à l'objet supprimé (vous perdez ainsi en quelque sorte l'historique)

Remarques:

R1. Il n'est pas nécessaire que des champs reliés aient un même nom. C'est le contenu qui fait foi.

R2. Les tables sources de la relation doivent avoir une clé primaire avec un index unique.

R3. La relation doit se baser sur le champ de la clé primaire (respectivement de l'index unique).

Lisez également le contenu de votre support et l'aide électronique intégrée à MS Access au sujet des relations pour avoir un complément d'informations.

Evidemment, habituellement il n'y a pas (du moins il ne devrait pas avoir) de données dans les tables lorsque l'on crée les relations !!!

Entrons maintenant un peu plus dans les détails au niveau des relations (partie théorique un peu barbante mais nécessaire).

Dans un premier temps, voici un rappel des conditions à satisfaire pour créer des relations:

1. La clé primaire est le champ de la table qui sert d'identifiant à un enregistrement. Il s'agit d'un champ indexé sans doublon.

2. La clé étrangère est le champ de la table qui servira à matérialiser la relation avec la clé primaire. Il doit être de même taille et de même type de données que la clé primaire à laquelle il est lié. Il est utile de lui donner le même nom que celui porté par la clé primaire correspondante.

3. Pour créer une relation de un à un, il faut impérativement lier des champs ayant leur propriété "indexé" à "oui-sans doublon".

4. Pour créer une relation de un à plusieurs, il suffit de lier les champs entre eux. D'habitude, la relation se fait d'une clé primaire vers une clé étrangère.

5. Pour créer une relation de plusieurs-à-plusieurs il faut créer une troisième table (dont le besoin et le contenu se définissent tout à fait naturellement), appelée une "table de jonction" (au fait c'est une table normale mais bon…), et lui ajouter des champs, clés étrangères, possédant les mêmes définitions que les champs clé primaire de chacune des deux autres tables.

6. Nous créons des clés primaires combinées dans des tables de transitions ce qui assure par exemple qu'un couple de champs soit unique.

Créez maintenant les relations suivantes entre les tables que vous avez pour l'instant par rapport au schéma ci-dessous (votre formateur vous expliquera les tenants et aboutissants dans les détails).

Attention, des erreurs ont exprès été glissées dans ce schéma (erreurs de nommage des champs, d'intégrité, de position de clé primaire, d'index, etc.) et… n'oubliez pas que vous n'avez pas encore toutes relativement à la finalité du cours donc ne paniquez pas si vous n'avez quelque chose d'identique à ce qui est montré dans la figure !!!

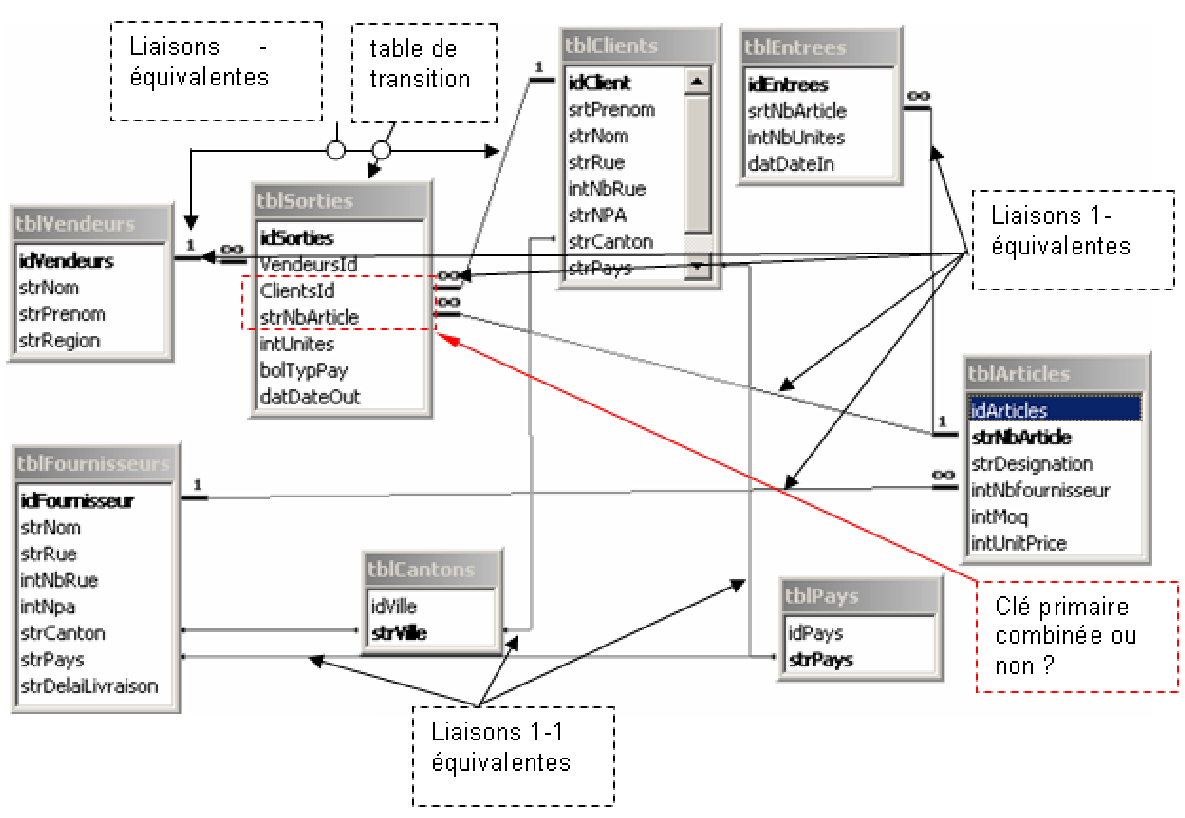
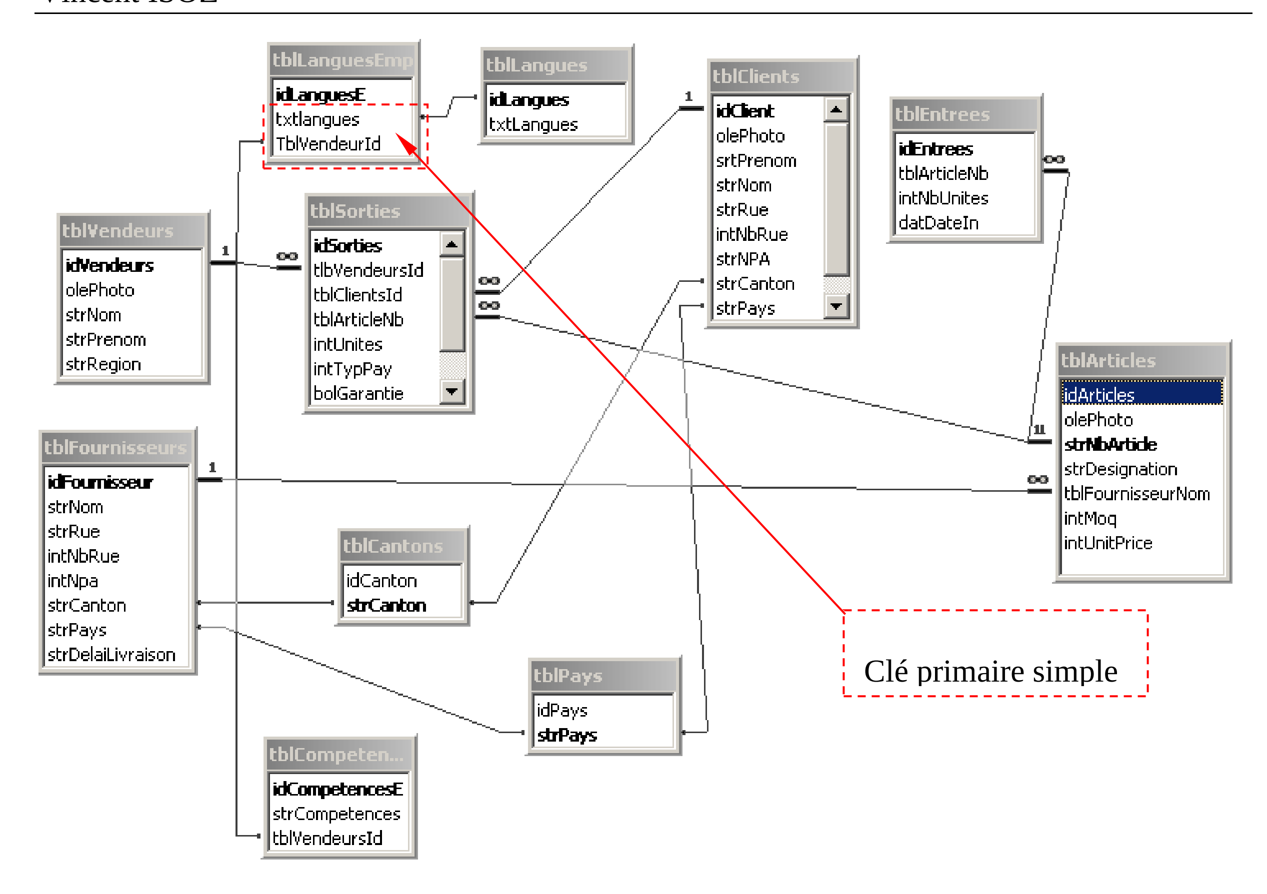


Figure 6 Schéma (simple) de la BDD

Remarques:

R1. D'une formation à l'autre, ce schéma peut changer un peu (votre formateur aime bien parfois varier ses cours…). Par exemple avec de bons participants en deux jours nous arrivons à un schéma relationnel comme celui de la page suivante:



R2. Certaines erreurs ont aussi été exprès introduites dans le schéma ci-dessus afin que les participants mettent en application leurs connaissances du cours initiation.

Pour vous exercer, vous pouvez tenter de répondre à ces questions (elles sont traitées dans les détails lors du cours):

1. Pourquoi la clé primaire de la table *tblArticles* n'a pas été mise sur le champ *idArticle*?
2. Où faudrait-il activer l'intégrité référentielle dans ce schéma ?
3. La mise-à-jour en cascade a-t-elle alors un sens pour les tables reliées cette clé primaire *strCodeArticle*?
4. Et la mise-à-jour en cascade pour les autres relations ?
5. Quand faut-il activer la suppression en cascades des champs ?
6. Comment pouvons-nous rendre un champ unique sans utiliser de clé primaire (VBA?, index?, etc.)???
7. Comment rendre le prénom et le nom d'un individu dans la table *tblClients* unique?
8. Faut-il mettre un index combiné dans la table *tblLanguesEmp* ou pas ? Pourquoi ?

Ce schéma est extrêmement simple et il est aussi extrêmement rare d'avoir quelque chose d'aussi petit et simple dans MS Access, une bonne moyenne (nivelée vers le bas) serait plutôt du type suivant (voir page suivante):

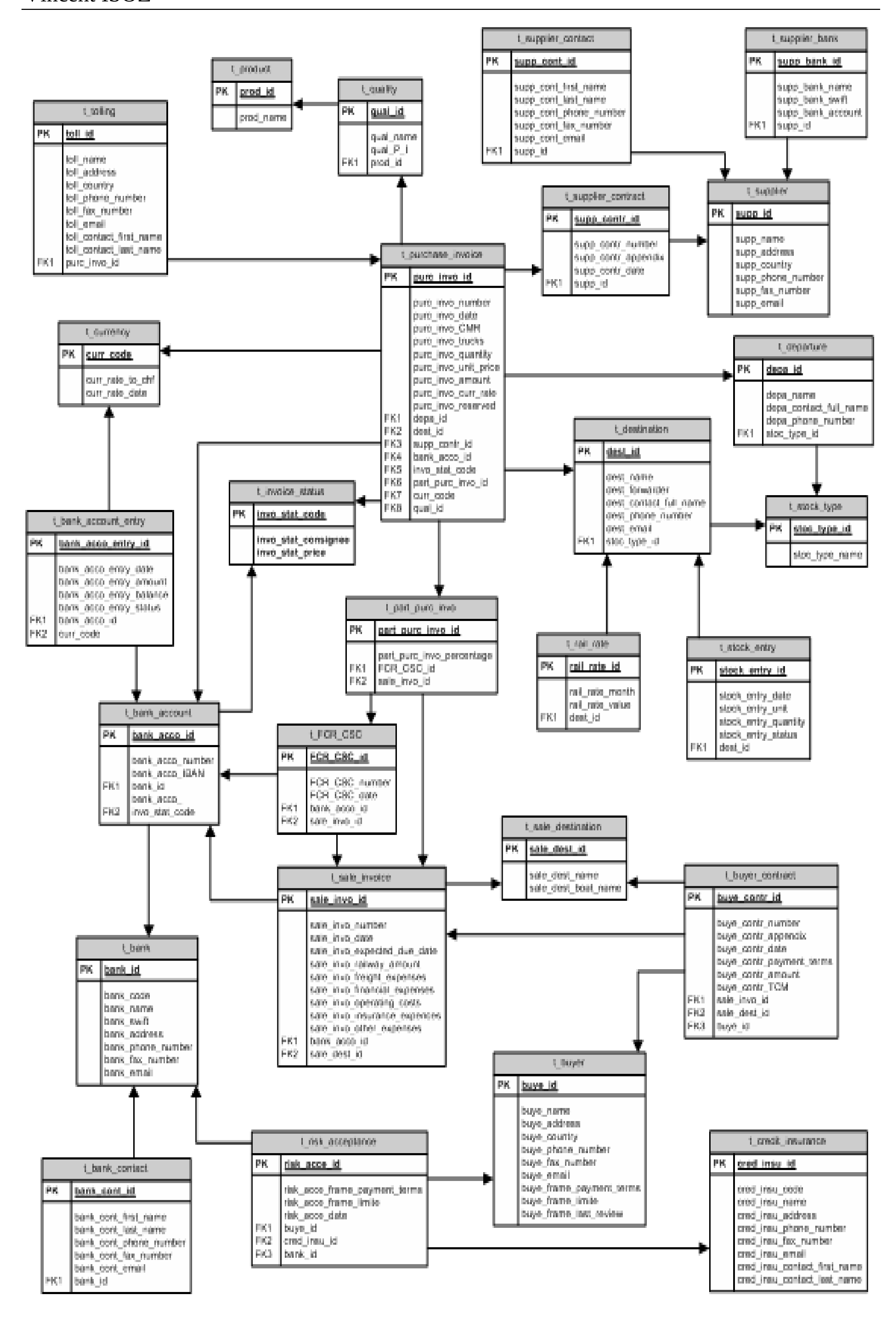

t_product
PK prod_id
prod_name
t_quality
PK qual_id
qual_name
qual_P_i
FK1 prod_id
t_tolling
PK toll_id
toll_name
toll_address
toll_country
toll_phone_number
toll_fax_number
toll_email
toll_contact_first_name
toll_contact_last_name
FK1 purc_invo_id
t_supplier_contact
PK supp_cont_id
supp_cont_first_name
supp_cont_last_name
supp_cont_phone_number
supp_cont_fax_number
supp_cont_email
FK1 supp_id
t_supplier_bank
PK supp_bank_id
supp_bank_name
supp_bank_swift
supp_bank_account
FK1 supp_id
t_supplier_contract
PK supp_contr_id
supp_contr_number
supp_contr_appendix
supp_contr_date
FK1 supp_id
t_supplier
PK supp_id
supp_name
supp_address
supp_country
supp_phone_number
supp_fax_number
supp_email
t_purchase_invoice
PK purc_invo_id
purc_invo_number
purc_invo_date
purc_invo_CMR
purc_invo_trucks
purc_invo_quantity
purc_invo_unit_price
purc_invo_amount
purc_invo_curr_rate
purc_invo_reserved
FK1 depa_id
FK2 dest_id
FK3 supp_contr_id
FK4 bank_acco_id
FK5 invo_stat_code
FK6 part_purc_invo_id
FK7 curr_code
FK8 qual_id
t_currency
PK curr_code
curr_rate_to_chf
curr_rate_date
t_departure
PK depa_id
depa_name
depa_contact_full_name
depa_phone_number
FK1 stoc_type_id
t_destination
PK dest_id
dest_name
dest_forwarder
dest_contact_full_name
dest_phone_number
dest_email
FK1 stoc_type_id
t_stock_type
PK stoc_type_id
stoc_type_name
t_invoice_status
PK invo_stat_code
invo_stat_consignee
invo_stat_price
t_bank_account_entry
PK bank_acco_entry_id
bank_acco_entry_date
bank_acco_entry_amount
bank_acco_entry_balance
bank_acco_entry_status
FK1 bank_acco_id
FK2 curr_code
t_part_purc_invo
PK part_purc_invo_id
part_purc_invo_percentage
FK1 FCR_CSC_id
FK2 sale_invo_id
t_rail_rate
PK rail_rate_id
rail_rate_month
rail_rate_value
FK1 dest_id
t_stock_entry
PK stock_entry_id
stock_entry_date
stock_entry_unit
stock_entry_quantity
stock_entry_status
FK1 dest_id
t_bank_account
PK bank_acco_id
bank_acco_number
bank_acco_IBAN
FK1 bank_id
bank_acco_
FK2 invo_stat_code
t_FCR_CSC
PK FCR_CSC_id
FCR_CSC_number
FCR_CSC_date
FK1 bank_acco_id
FK2 sale_invo_id
t_sale_destination
PK sale_dest_id
sale_dest_name
sale_dest_boat_name
t_sale_invoice
PK sale_invo_id
sale_invo_number
sale_invo_date
sale_invo_expected_due_date
sale_invo_railway_amount
sale_invo_freight_expenses
sale_invo_financial_expenses
sale_invo_operating_costs
sale_invo_insurance_expenses
sale_invo_other_expenses
FK1 bank_acco_id
FK2 sale_dest_id
t_buyer_contract
PK buye_contr_id
buye_contr_number
buye_contr_appendix
buye_contr_date
buye_contr_payment_terms
buye_contr_amount
buye_contr_TCM
FK1 sale_invo_id
FK2 sale_dest_id
FK3 buye_id
t_bank
PK bank_id
bank_code
bank_name
bank_swift
bank_address
bank_phone_number
bank_fax_number
bank_email
t_buyer
PK buye_id
buye_name
buye_address
buye_country
buye_phone_number
buye_fax_number
buye_email
buye_frame_payment_terms
buye_frame_limite
buye_frame_last_review
t_risk_acceptance
PK risk_acce_id
risk_acce_frame_payment_terms
risk_acce_frame_limite
risk_acce_date
FK1 buye_id
FK2 cred_insu_id
FK3 bank_id
t_credit_insurance
PK cred_insu_id
cred_insu_code
cred_insu_name
cred_insu_address
cred_insu_phone_number
cred_insu_fax_number
cred_insu_email
cred_insu_contact_first_name
cred_insu_contact_last_name
t_bank_contact
PK bank_cont_id
bank_cont_first_name
bank_cont_last_name
bank_cont_phone_number
bank_cont_fax_number
bank_cont_email
FK1 bank_id

Figure 7 Exemple de BDD (taille standard)

Revenons à notre exemple de départ et détaillons-le:

Précisions sur l'intégrité référentielle:

L'intégrité référentielle est un mécanisme de vérification qui s'assure que chaque clé étrangère est en correspondance avec sa clé primaire. Non seulement il vérifie l'intégrité des données écrites dans la clé étrangère, mais encore, il maintient cette intégrité en cas de modification ou de suppression de la clé primaire.

En conséquence, lorsque l'intégrité référentielle est appliquée sur une relation, on ne peut pas trouver, dans la clé étrangère, de données autres que celles contenues dans la clé primaire correspondante.

Dès lors, bien que toutes les données figurant dans la table contenant la clé étrangère soient en relations avec les données de la table contenant la clé primaire, l'inverse n'est pas exact.

Si la "Mise à jour en cascade" et la "Suppression en cascade" ne sont pas actives, il n'est pas donné à l'utilisateur la possibilité de supprimer ou de mettre à jour AUTOMATIQUEMENT le champ de la table d'origine étant déjà en relation avec des champs de la table étrangère. Il sera toujours possible de le faire par automatisme ou programmation.

Testez, comme exercice:

1. L'intégrité référentielle
2. L'intégrité référentielle avec la mise à jour en cascade

3. L'intégrité référentielle avec la suppression en cascade

Au hasard, un des participants ira au poste du formateur pour démontrer le bon fonctionnement de ces outils.

5.7 Relation circulaire (auto-liaison)

La relation circulaire (dite aussi "auti-jointure" ou "jointure réflexive") est un cas typique du modèle conceptuel de données (MCD). Il s'agit au fait d'une relation entre une table et elle-même!

Malheureusement, il semblerait que cette fonctionnalité ne marche plus comme elle devrait depuis Access 2007. Ce qui est une énorme perte étant données les possibilités qui étaient offertes par les auto-liaisons. En attendant, le lecteur pourra sauter ce sous-chapitre.

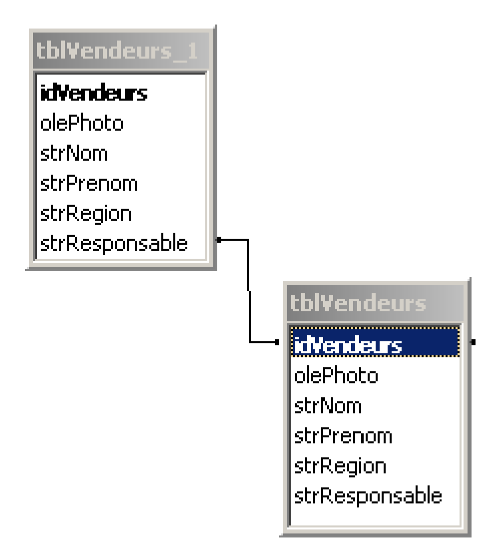
Illustrons ce concept par un exemple.

Considérons la table suivante:



Nous souhaitons définir pour chaque vendeur, un vendeur responsable (une structure d'organigramme d'entreprise en d'autres termes…).

Pour faire cela, ajoutons à cette table un champ *strResponsable* avec un champ de type *Integer*. Ensuite dans la fenêtre des relations, ajoutons une deuxième fois la table *tblVendeurs* (ce qui s'appelle techniquement un "alias de table"):



Remarque: Il est important de ne pas activer l'intégrité référentielle car il y a aura normalement toujours au moins un employé qui n'auras pas de supérieur.

Dans la table *tblVendeurs* allez dans les propriétés du champ *strResponsable* et activez les options suivantes:

Général	Liste de choix
Afficher le contrôle	Zone de liste
Origine source	Table/Requête
Contenu	
Colonne liée	1
Nbre colonnes	2
En-têtes colonnes	Non
Largeurs colonnes	0;2.54 cm

Ensuite éditez la requête de liste (le champ **Contenu** ci-dessus) de choix *strResponsable* en y écrivant:

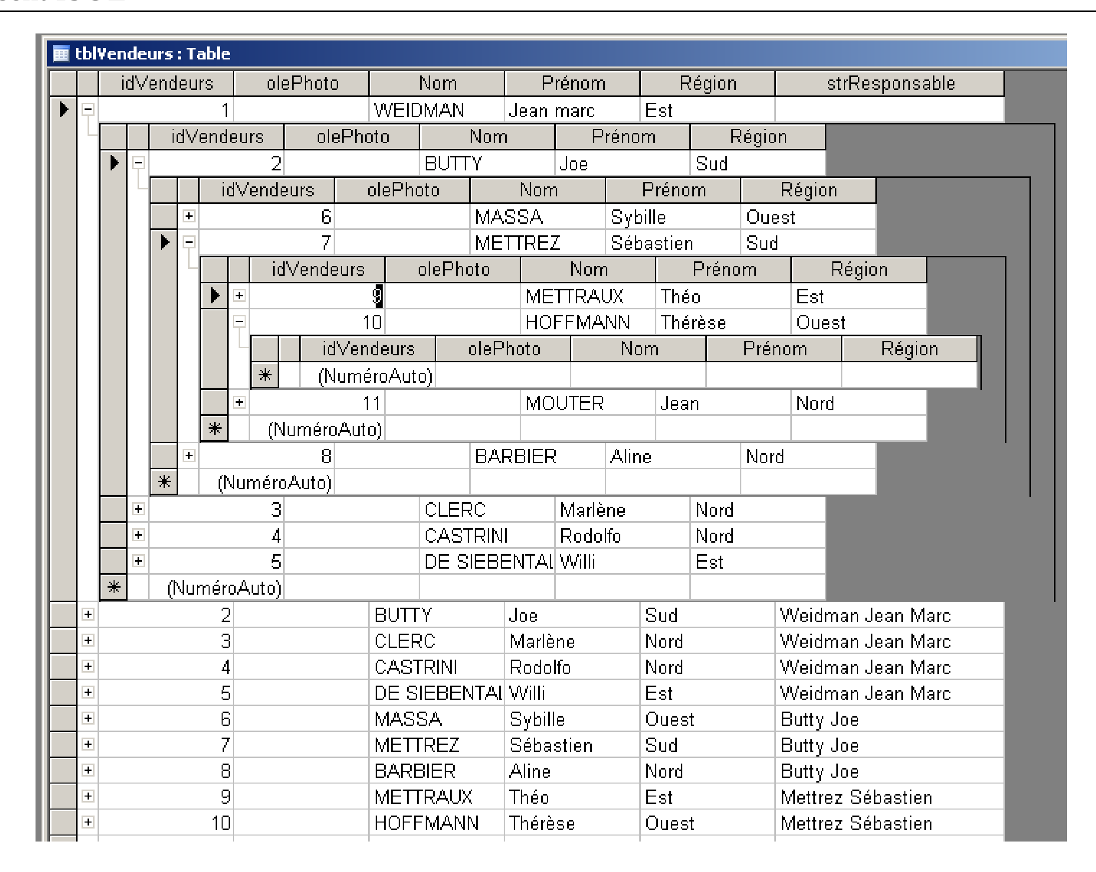
SELECT [idVendeurs], [strNom] & " " & [strPrenom] AS Resp FROM tblVendeurs

De retour dans votre table *tblVendeurs*, vous aurez alors le résultat attendu:

		idVendeurs	olePhoto	Nom	Prénom	Région	strResponsable
	+	1		WEIDMAN	Jean marc	Est	
	+	2		BUTTY	Joe	Sud	Weidman Jean Marc
	+	3		CLERC	Marlène	Nord	Weidman Jean Marc
	+	4		CASTRINI	Rodolfo	Nord	Weidman Jean Marc
	+	5		DE SIEBENTAI	Willi	Est	Weidman Jean Marc
	+	6		MASSA	Sybille	Ouest	Butty Joe
	+	7		METTREZ	Sébastien	Sud	Butty Joe
	+	8		BARBIER	Aline	Nord	Butty Joe
	+	9		METTRAUX	Théo	Est	Mettrez Sébastien
	+	10		HOFFMANN	Thérèse	Ouest	Mettrez Sébastien
	+	11		MOUTER	Jean	Nord	Mettrez Sébastien
*		(NuméroAuto)					

Castrini Rodolfo
De Siebental Willi
Massa Sybille
Mettrez Sébastien
Barbier Aline
Mettraux Théo
Hoffmann Thérèse
Mouter Jean

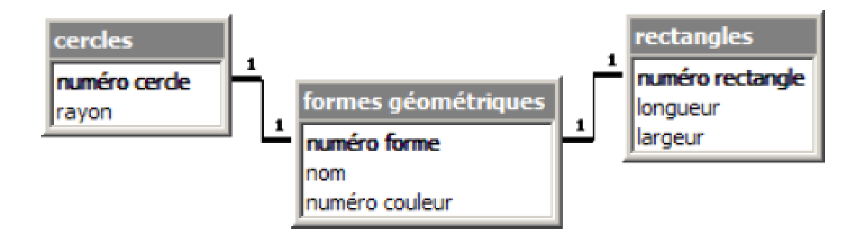
et si nous cliquons sur les petits + en sélectionnant la table *tblVendeurs* nous avons alors un visuel de l'organigramme (cellulaire) de vendeurs:



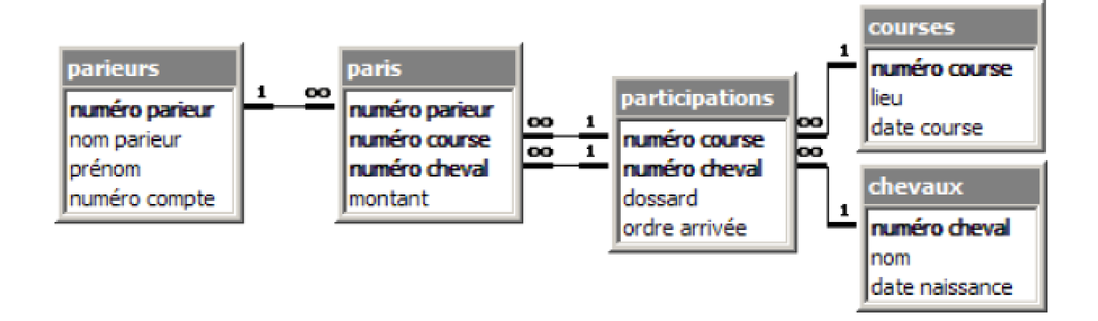
5.8 Relations d'héritage et composites

Il existe encore d'autres types de relations comme les "relations composites" ou "relations d'héritage" (utilisées dans la modélisation objet) ainsi que les relations de "sous-numérotation" mais elles nécessitent des connaissances telles pour les appliquer tant un cas concret que le développeur amateur ne travaille dès lors plus sous MS Access normalement et se trouve dès lors être plus un ingénieur qu'un utilisateur bureautique.

Exemple de relation d'héritage pour la culture générale du lecteur:



Exemple de relation composite pour la culture générale du lecteur:



5.9 Index simples et combinés

Pour que MS Access puisse accéder directement à des enregistrements sans devoir lire chaque enregistrement du début à la fin d'une table (par le moteur JET/Microsoft Jet Database Engine), les index peuvent être utilisés. L'indexation accélère ainsi la recherche et le tri d'enregistrements.

Un index dans une base de données est comparable à l'index d'un livre dans lequel les numéros de pages sont indiqués pour certaines expressions. Si vous recherchez une expression dans un livre, vous avez les possibilités suivantes:

1. Lire chaque page (1, 2, 3, ..) jusqu'à ce que vous ayez trouvé l'expression recherchée.

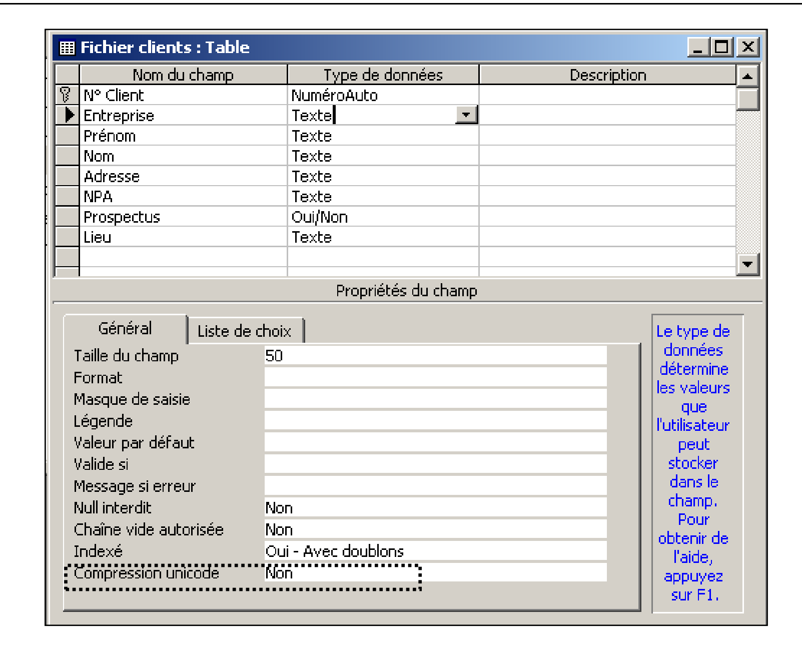
2. Consulter l'index pour trouver le numéro de la page où se trouve l'expression recherchée.

Derrière chaque table de données se cache un index dans lequel Access fait référence à chaque enregistrement d'une ligne donnée. A chaque fois qu'une requête sera effectuée dans la table, celle-ci sera plus ou moins longue en fonction du type d'index et non du contenu de la table.

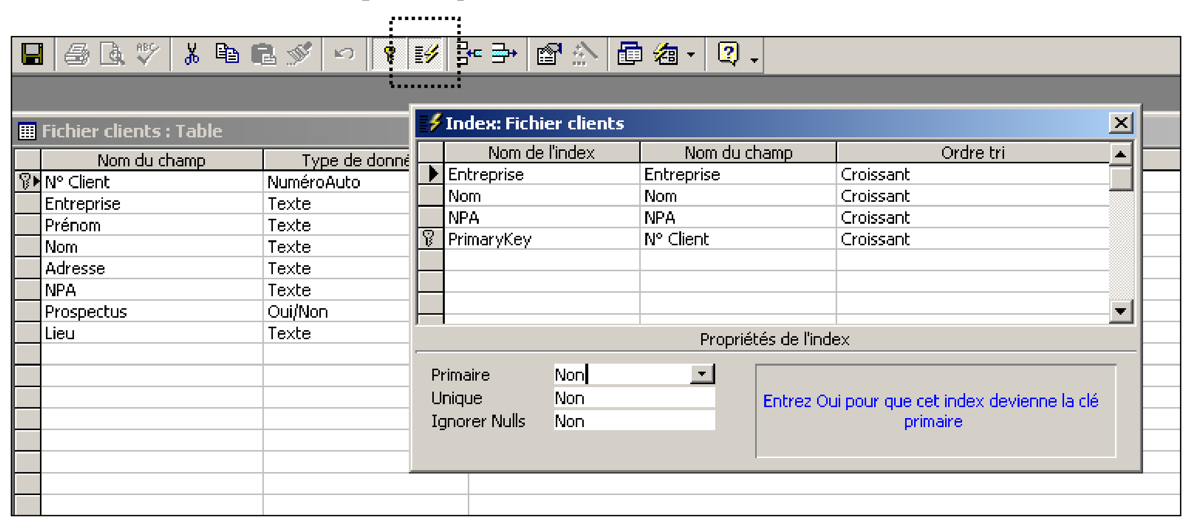
Vous devriez créer un index pour tous les champs ou combinaisons de champs avec lesquels vous rechercherez ou trierez souvent. Et notamment sur les champs clés primaires et clés étrangères.

Considérez toujours que chaque index provoque un ralentissement de la saisie ou de la modification des données. En arrière-plan, MS Access doit automatique mettre à jour toutes les tables d'index. En plus, chaque table d'index occupe de la place sur le disque dur.

Un index est créé automatiquement dès que vous définissez une clé primaire. Sinon, pour créer (ou changer) un index, il suffit d'aller dans une table en *mode création* et dans le champ *Indexé* choisir *Non, Oui (avec doublons)* ou *Oui (sans doublons).*



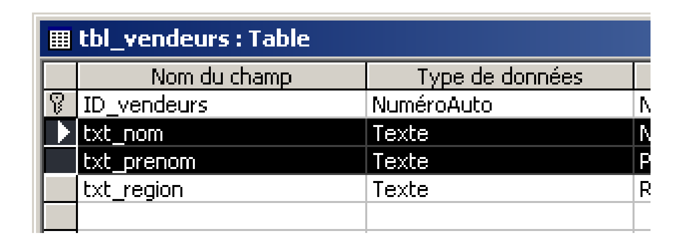
Ouvrez la table *tblClients*, basculez en *Mode création*. Cliquez sur le bouton mis en évidence ci-dessous et observez les options qui vous sont offertes:



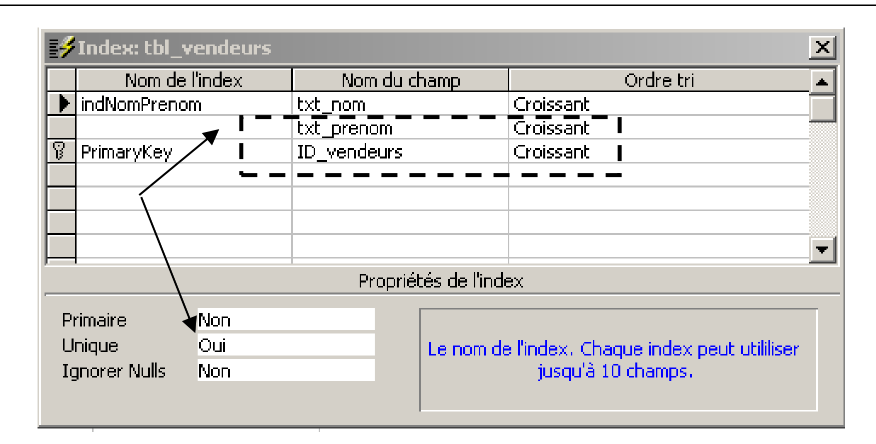
Remarque: Lisez également le contenu de votre support et l'aide électronique intégrée à MS Access au sujet des index.

Testez, comme exercice, avec la table *tblClients* le bon fonctionnement d'une clé simple sur le champ *strNom* et d'une clé combinée sur les champs *strNom* et *strPrenom*.

Méthode typique avec le *Nom* et le *Prénom* de la table *tblVendeurs:*



Cliquez sur le bouton de gestion des index et créez la structure suivante:



Le fait de ne pas mettre d'index à la deuxième ligne a pour effet (au même titre que pour les macros que nous verrons plus loin) de cumuler l'index, et ainsi de le combiner, entre *strNom* et *strPrenom.*

6 Formulaires (simples)

Les formulaires sont des objets importants dans MS Access. Effectivement, ce sont eux qui vont permettre à l'utilisateur de la base de données de gagner du temps relativement à ces besoins quotidiens. De plus, il est connu que lorsqu'une application est conviviale, elle est beaucoup plus appréciée par les utilisateurs.

Remarques:

R1. Les formulaires (et la partie "configuration de l'esthétique des formulaires") sont aussi traités dans le chapitre "Interfaçage de l'application" (voir page 254) de ce support.

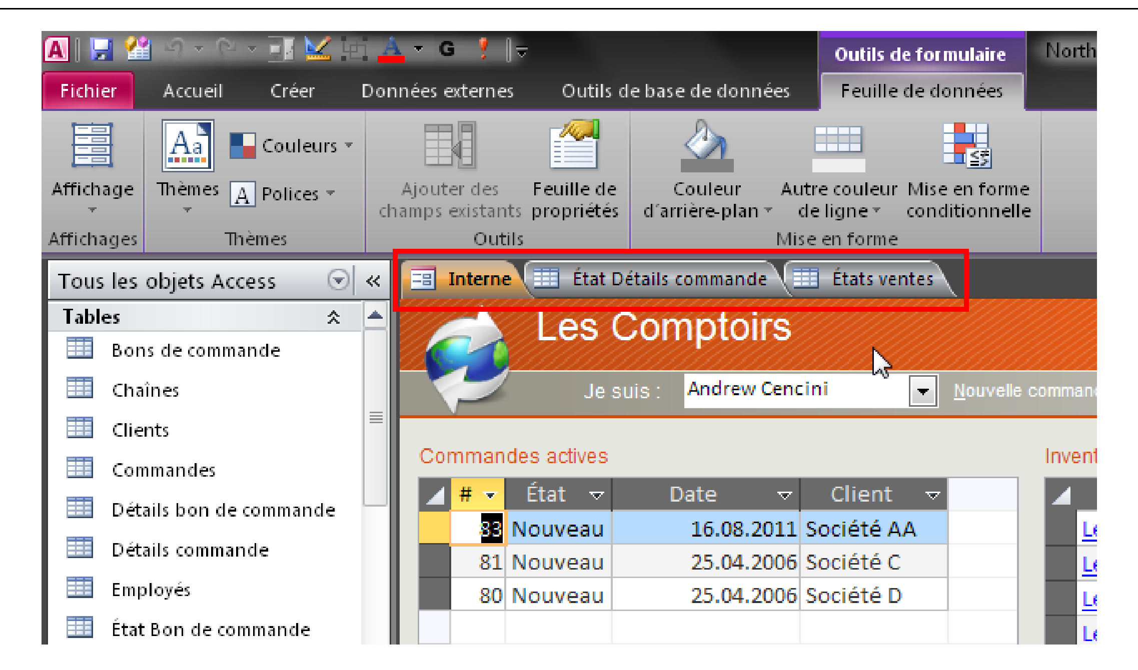
R2. Ne créez jamais de formulaires en les mettant en mode "pleine fenêtre" ce n'est pas du tout une bonne méthode de travail pour des raisons de compatibilité et de veille technologique.

R3. Essayez toujours de faire des formulaires (interfaces) aussi petits que possibles car de plus en plus d'entreprises équipent leurs employés en PDA équipés de Windows CE et d'un runtime MS Access. Nous verrons tout à la fin du document à quoi ressemble notre base de données sur un SmartPhone Qtek 9090 équipé du logiciel *DB Anywhere* (www.handango.com) permettant de synchroniser et convertir les bases MS Access au format CDB.

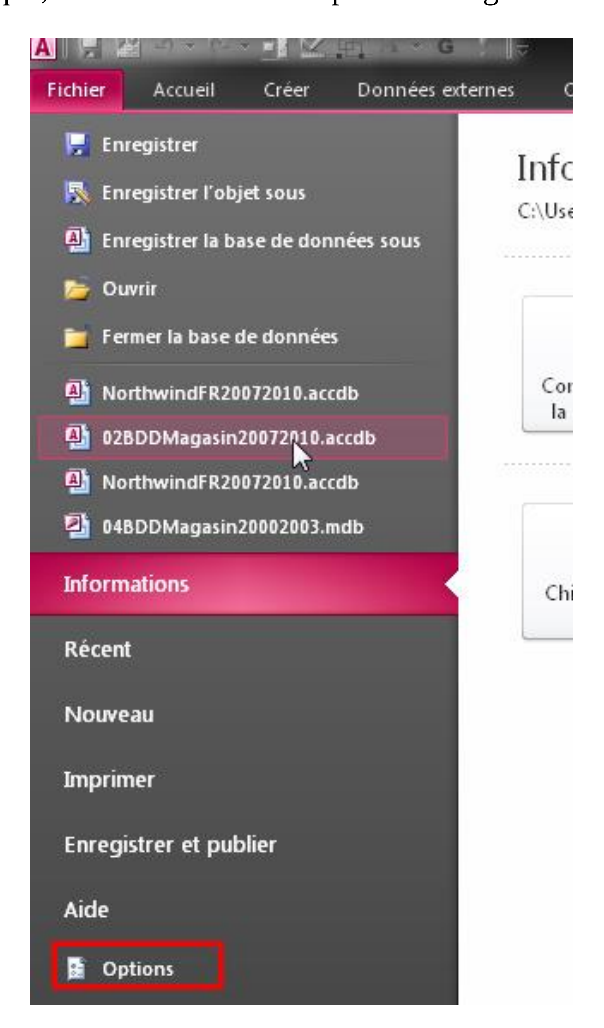


6.1 Choix de la méthode d'affichage des formulaires

Depuis Microsoft Access 2007 les formulaires s'affichent en mode non-modal en tant qu'onglets dans la fenêtre principale de Microsoft Access comme le montre la capture d'écran de la base *Les Comptoirs* ci-dessous:



Le côté pratique des ces onglets c'est qu'ils que même si un formulaire est devant un autre, nous savons qu'il y en a derrière. Nous pouvons au même titre qu'à l'époque où nous avions les fenêtres des formulaires modales passer de l'un à l'autre par le raccourci **Ctrl+F6** mais à part cela, beaucoup de concepteurs n'aiment pas cette nouvelle présentation. Pour revenir alors à la version classique, il faut aller dans les options du logiciel:



Ensuite:

et libre à vous de faire le choix qui convient.

6.2 Filtre par formulaire

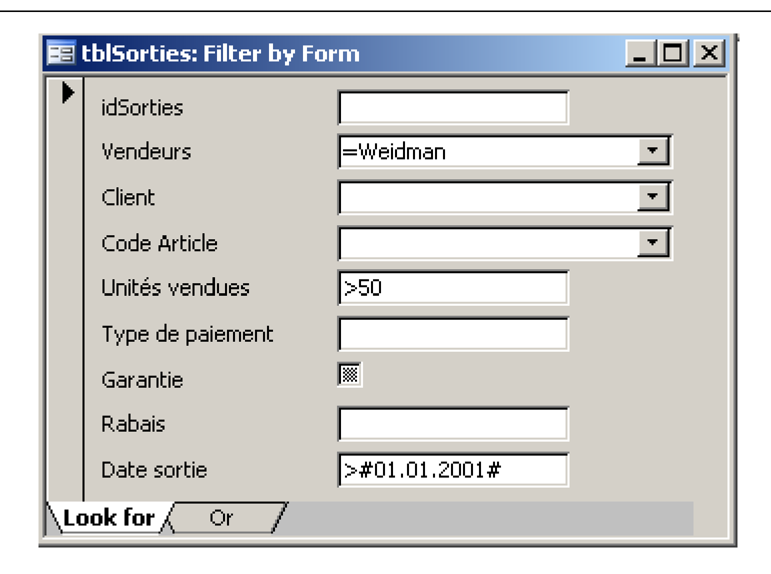
Attention!!!

1. La fonctionnalité décrite ci-dessous ne marche pas en mode Runtime ni en mode *.mde!!!

2. Le filtre par formulaire une fois activé, désactive tous les boutons présents sur le formulaire et cela ne semble pas pouvoir a priori être contourné (donc impossible de créer un bouton pour appliquer un filtre par formulaire ou pour charger une filtre d'une requête)

Nous allons maintenant voir un outil fort utile pour l'utilisateur de la base de données !!! Ouvrez le formulaire instantané *frmSorties*.

Cliquez sur le bouton pour activer le *Filtre par formulaire*. Définissez un ou plusieurs critères (observez l'onglet "Ou" se situant en-bas du formulaire !!!) tel que présenté ci-dessous (c'est une version un peu plus élaborée du formulaire *frmSorties* que vous n'avez pas nécessairement mais le principe est le même):

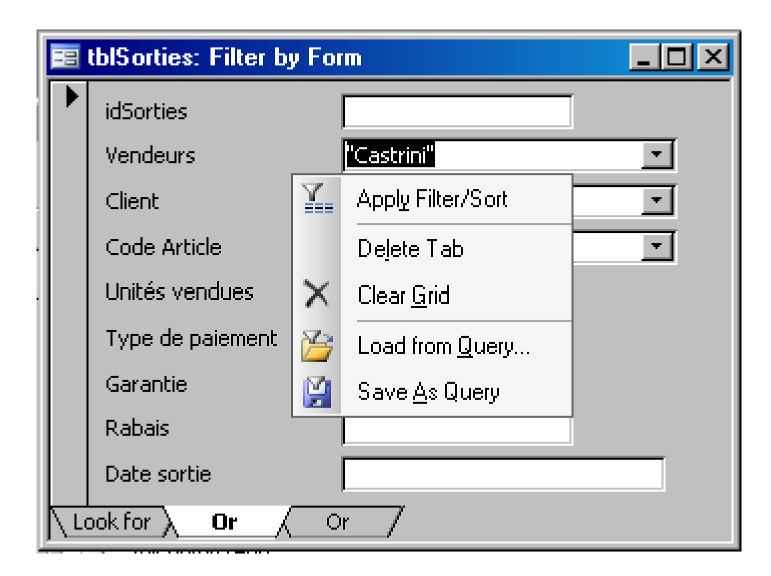


Activez ensuite le filtre en cliquant sur le bouton .

A chaque critère supplémentaire un nouvel onglet *OR* s'ajoute:

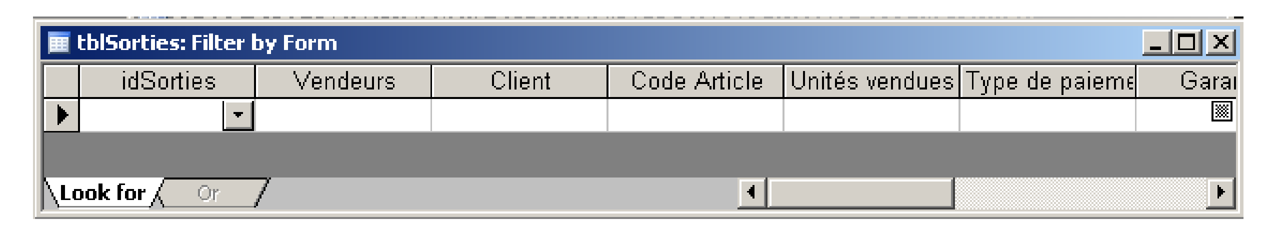


Si vous en avez plusieurs et que vous souhaitez en supprimer un en particulier, il suffit de faire un clic droit et choisir *Delete Tab*:



Vous verrez que le formulaire ne vous indique plus que les enregistrements qui satisfont aux critères choisis. Vous pouvez maintenant les parcourir et les modifier comme lorsque vous travaillez avec un formulaire normal.

Remarque: Ce "filtre par formulaire" comme son nom ne l'indique pas, fonctionne aussi à partir des tables directement (pour les personnes intéressées…):



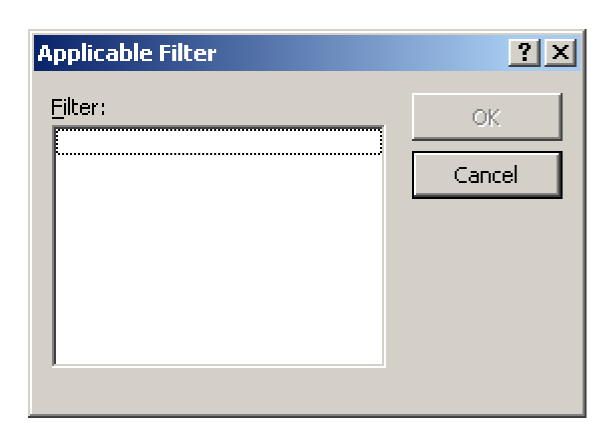
Un autre intérêt des filtres par formulaire est qu'il est possible de sauvegarder (avec leurs tris!) ceux-ci sans avoir connaissance du concept de "requête". Pour ce faire, une fois le filtre par formulaire créé, il suffit en haut à gauche de l'écran de cliquer sur le bouton:



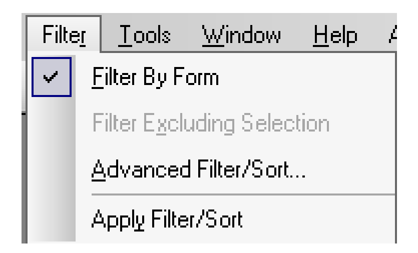
qui vous demandera sous quel nom vous souhaitez enregistrer les paramètres du filtre par formulaire en cours. Plus tard, (le lendemain, la semaine prochaine…) vous pourrez ainsi à tout moment rouvrir ce filtre enregistré en cliquant sur:



et en choisissant dans la liste qui se proposera à vous dans la boîte de dialogue suivante:



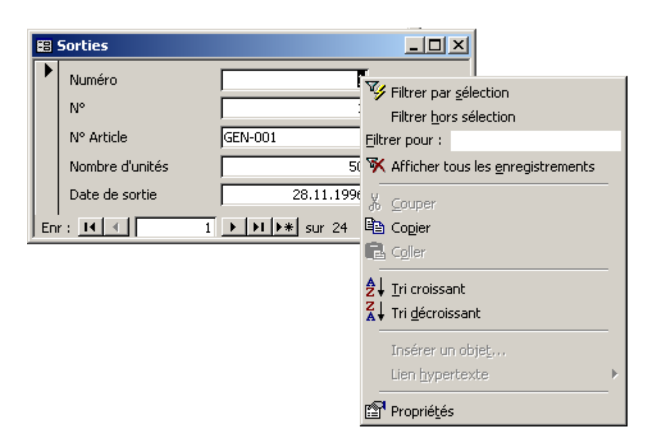
le filtre que vous souhaitez réutiliser s'il y en un! Mais attention!!!! vous perdrez les tris si vous chargez le filtre en mode *Filtre par formulaire* lorsque vous êtes dans une table. Si vous souhaitez contourner cette limitation, lorsque vous êtes dans le formulaire il faut passer par le menu:



et aller en mode *Advanced Filter/Sort…* et depuis là charger la requête précédemment enregistrée. Vous pourrez alors retrouver votre filtre et votre requête!

Il y a cependant une autre possibilité d'activer un filtre et de trier des données qu'en passant par un "Filtre par formulaire" mais il n'est quasiment jamais connu pas les utilisateurs lambda:

Sur un formulaire quelconque on peut par clique droit de la souris (si cette action a été autorisée par l'administrateur de la base de données) activer les options que nous avons déjà vues précédemment tel que l'on ait à l'écran la même chose que ce qui est représenté sur la figure de la page suivante.



La méconnaissance de cette fonction implique la nécessité de créer des formulaires spéciaux dans votre base de données faisant office de "Guide de l'utilisateur".

6.3 Formulaire en mode design

Créez un petit guide d'utilisateur sur le futur formulaire d'accueil de votre base de données (saisissez également votre nom, prénom et adresse e-mail pour le helpdesk dans un champ de légende).

A faire avec votre formateur.

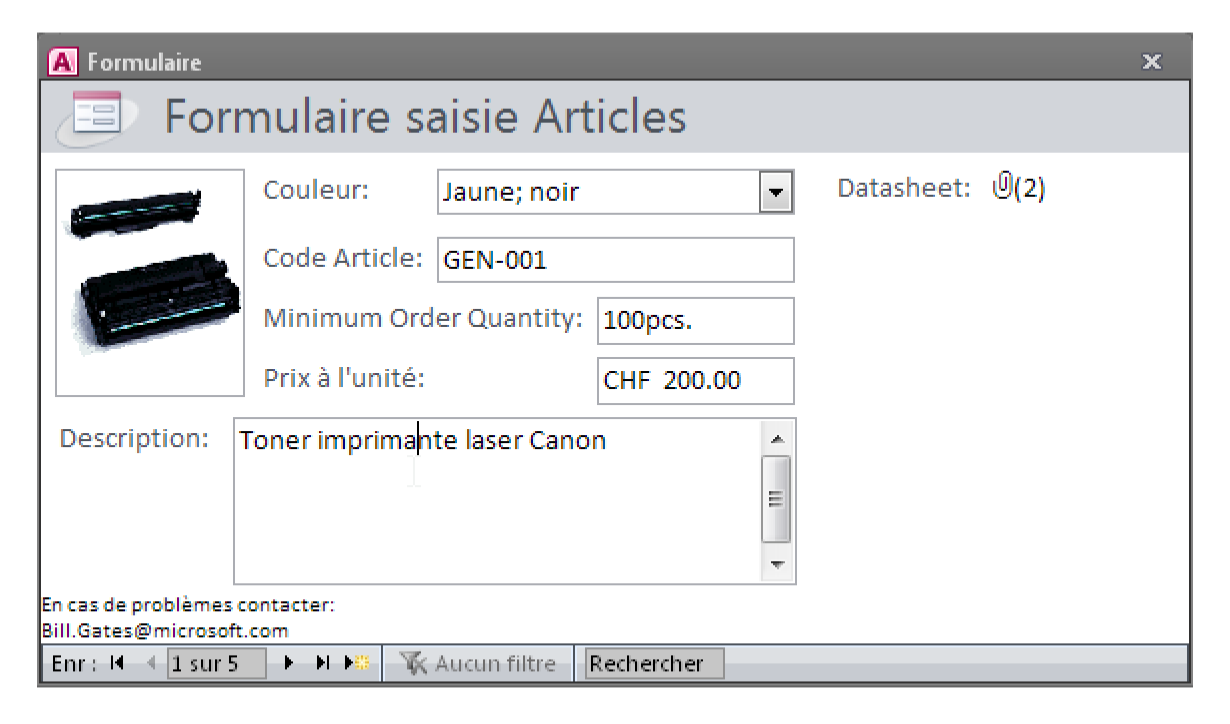
6.4 Création de boutons de formulaires

Si l'on revient cependant à notre filtre par formulaire, nous avons vu que le problème réside dans le fait que l'utilisateur de la base (en supposant que vous en êtes le responsable) ne sait pas nécessairement que cet outil (et tous ceux que nous avons vu précédemment) existe et qu'il est disponible dans la barre d'outils. C'est pourquoi, nous allons tout de suite voir comment ajouter des boutons sur le formulaire, qui permettront:

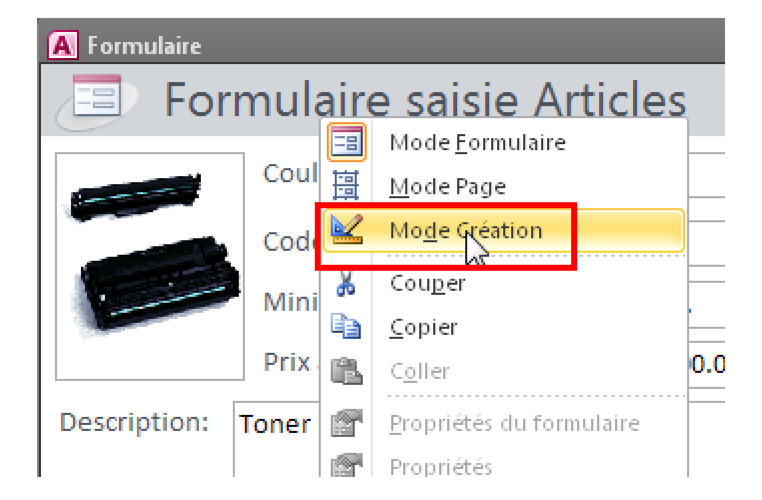
1. D'activer le filtrage des données (nous avons déjà fait un travail équivalent dans un des exercices précédents!).

2. De modifier les propriétés du filtre

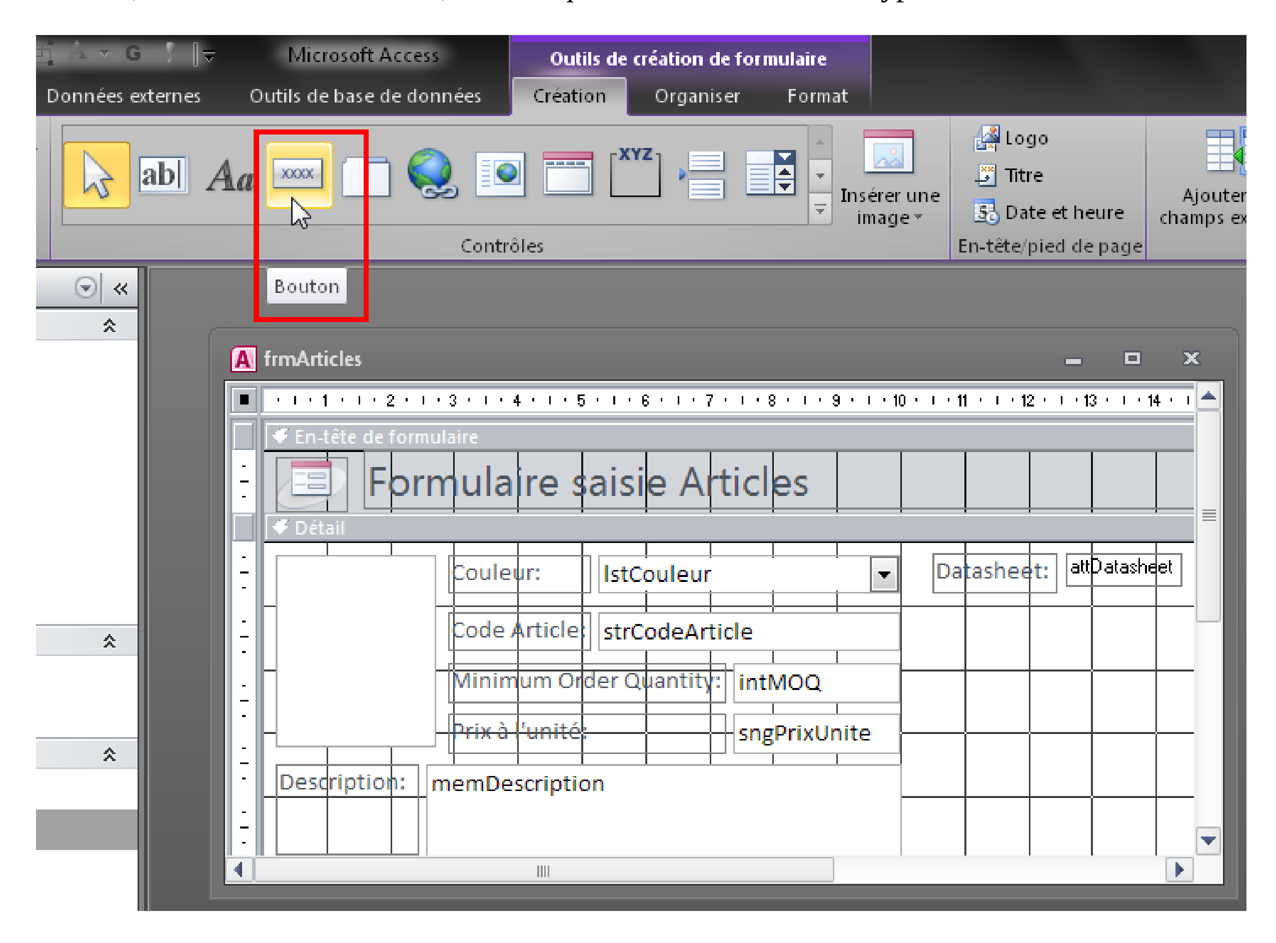
Pour l'exemple nous utiliserons le formulaire suivant:



Nous le passons en mode création:



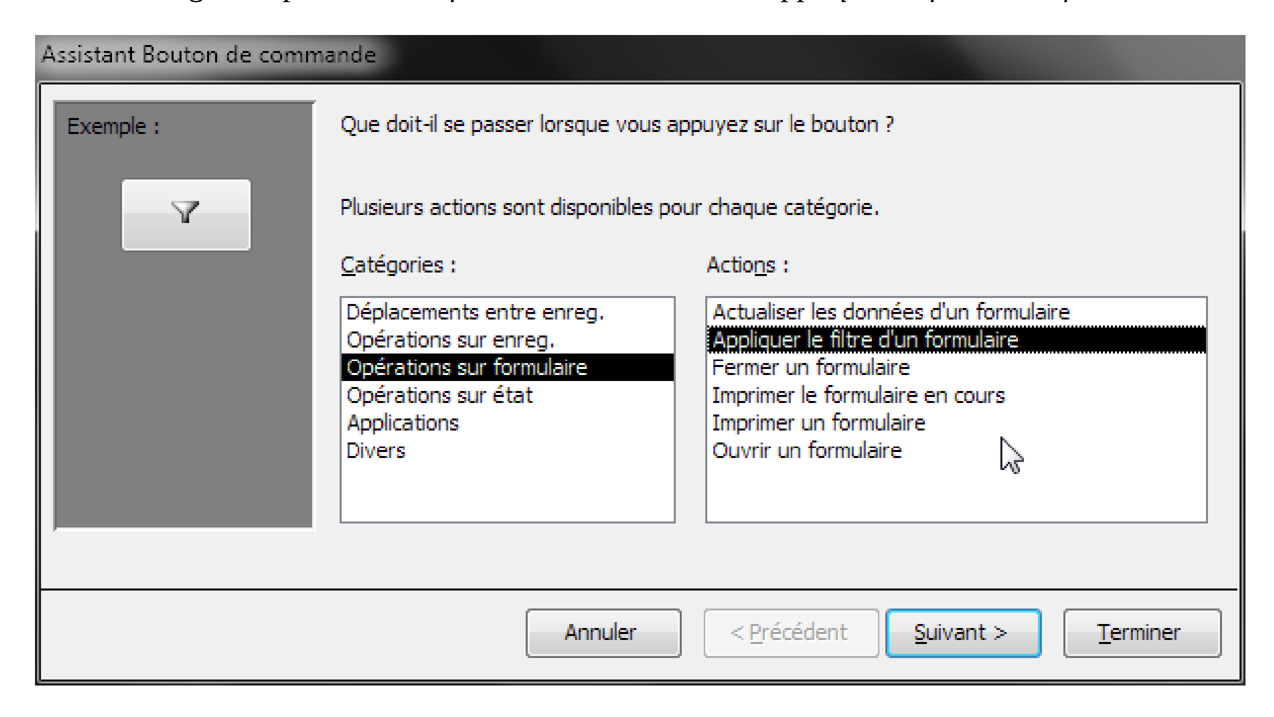
Ensuite, dans le ruban *Création*, nous cliquons sur le contrôle de type *Bouton*:



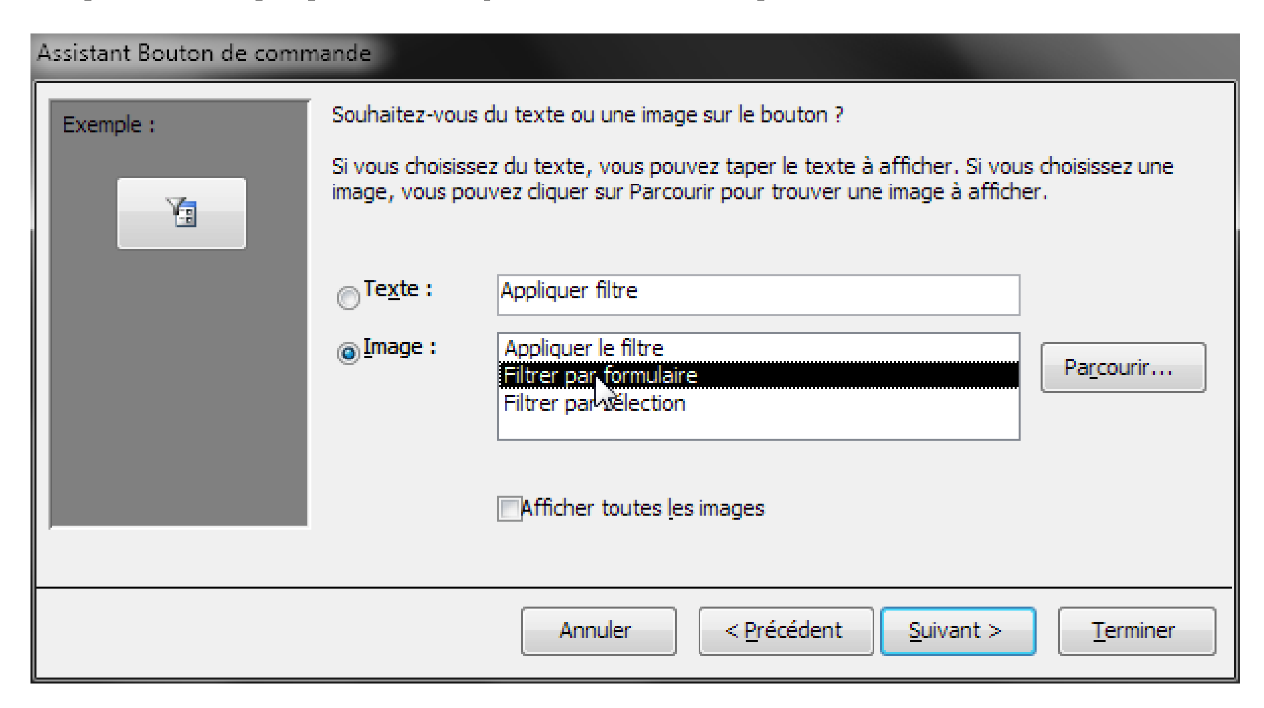
Ensuite nous dessinon un bout à l'endroit de notre choix:



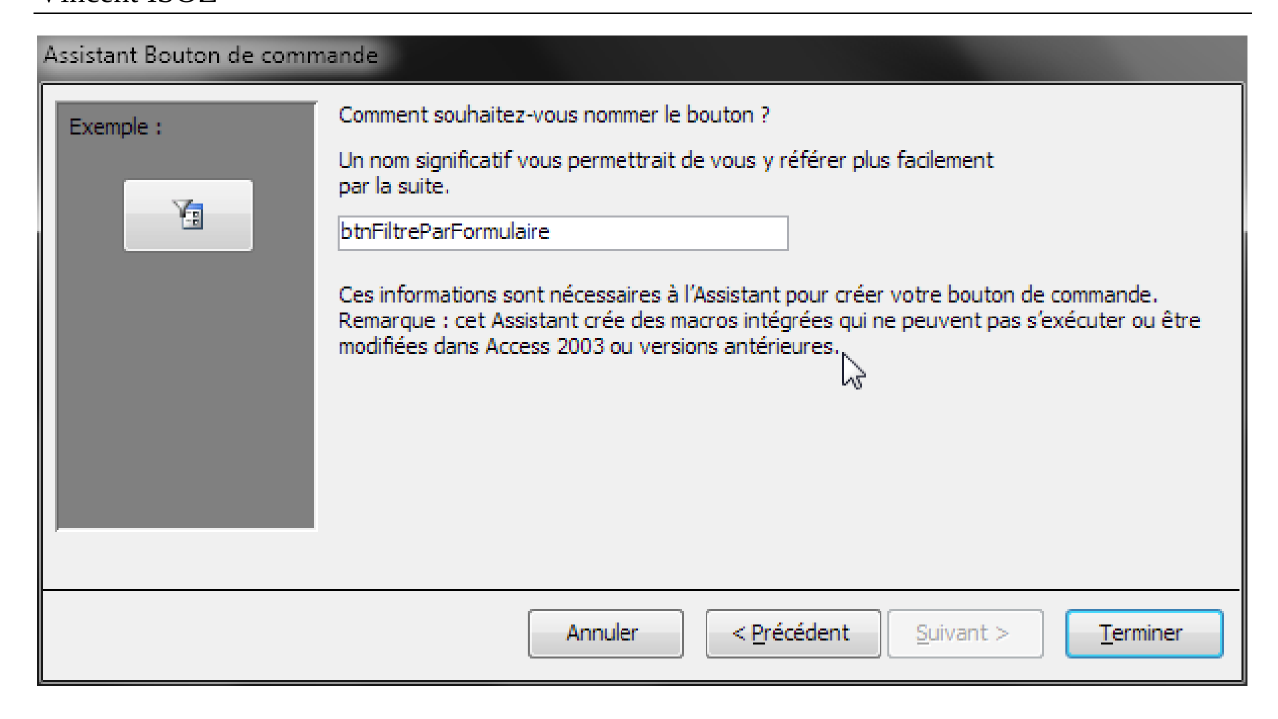
Une fois que nous lachons le bouton gauche de la souris un assistant vient et nous prenons alors la catégorie *Opération sur formulaire* avec l'action *Appliquer le filtre d'un formulaire*:



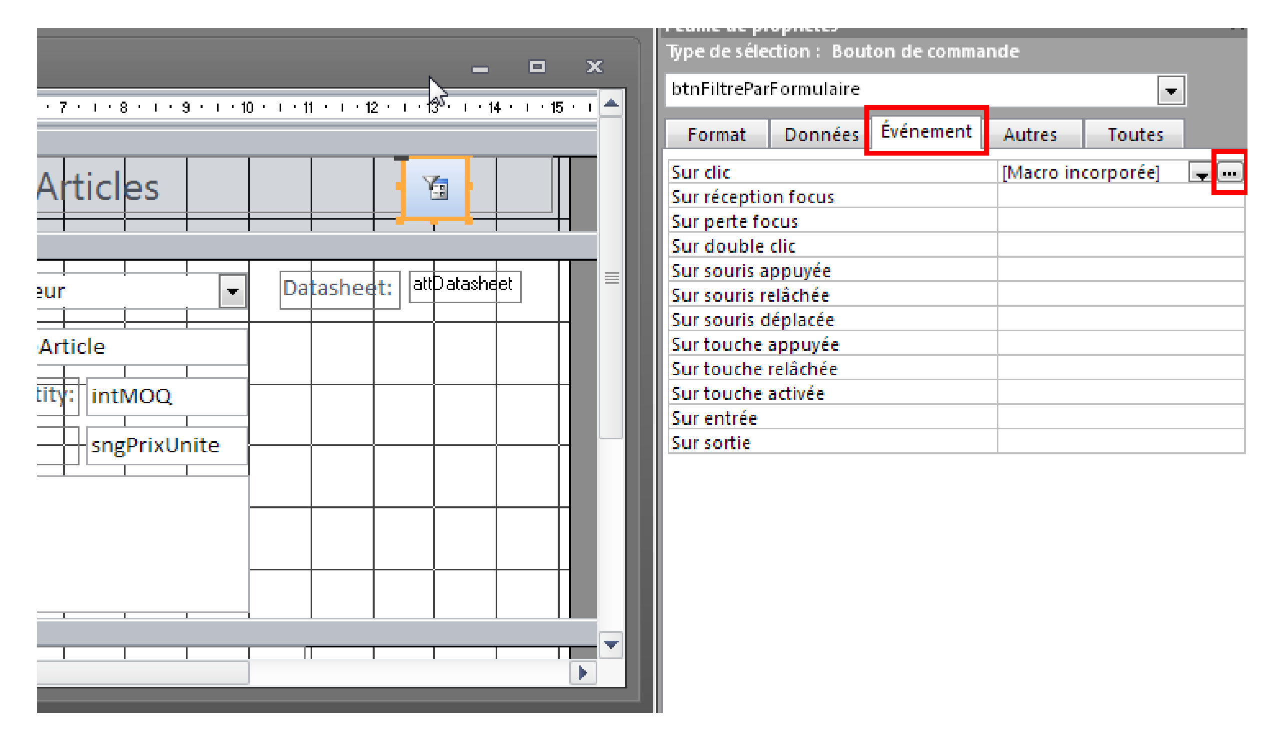
à la prochaine étape après avoir cliqué sur *Suivant*, nous prenons:



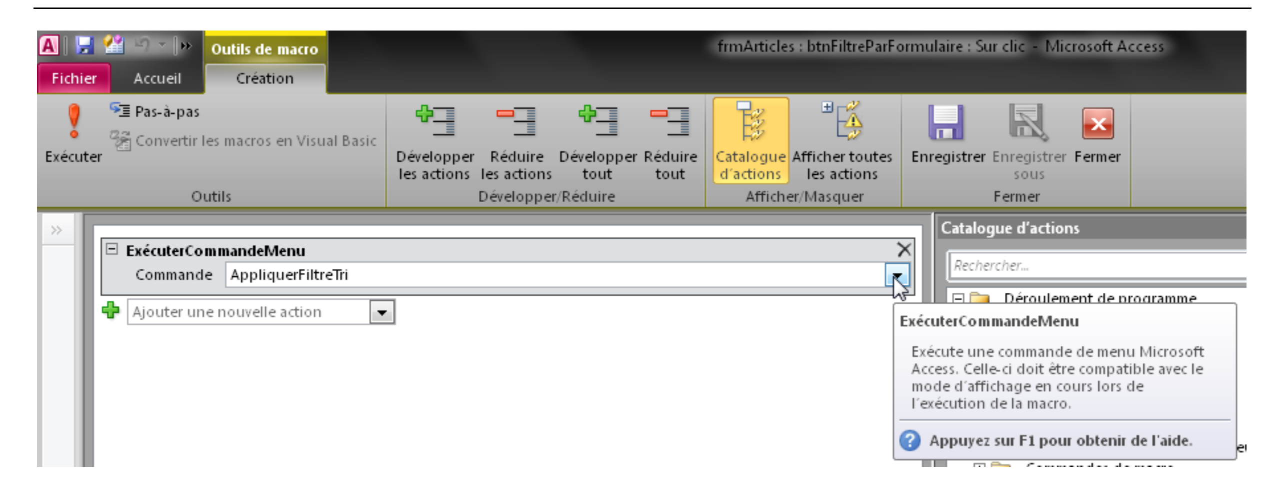
Nous cliquons ensuite sur *Suivant* et donnons un nom au bouton:



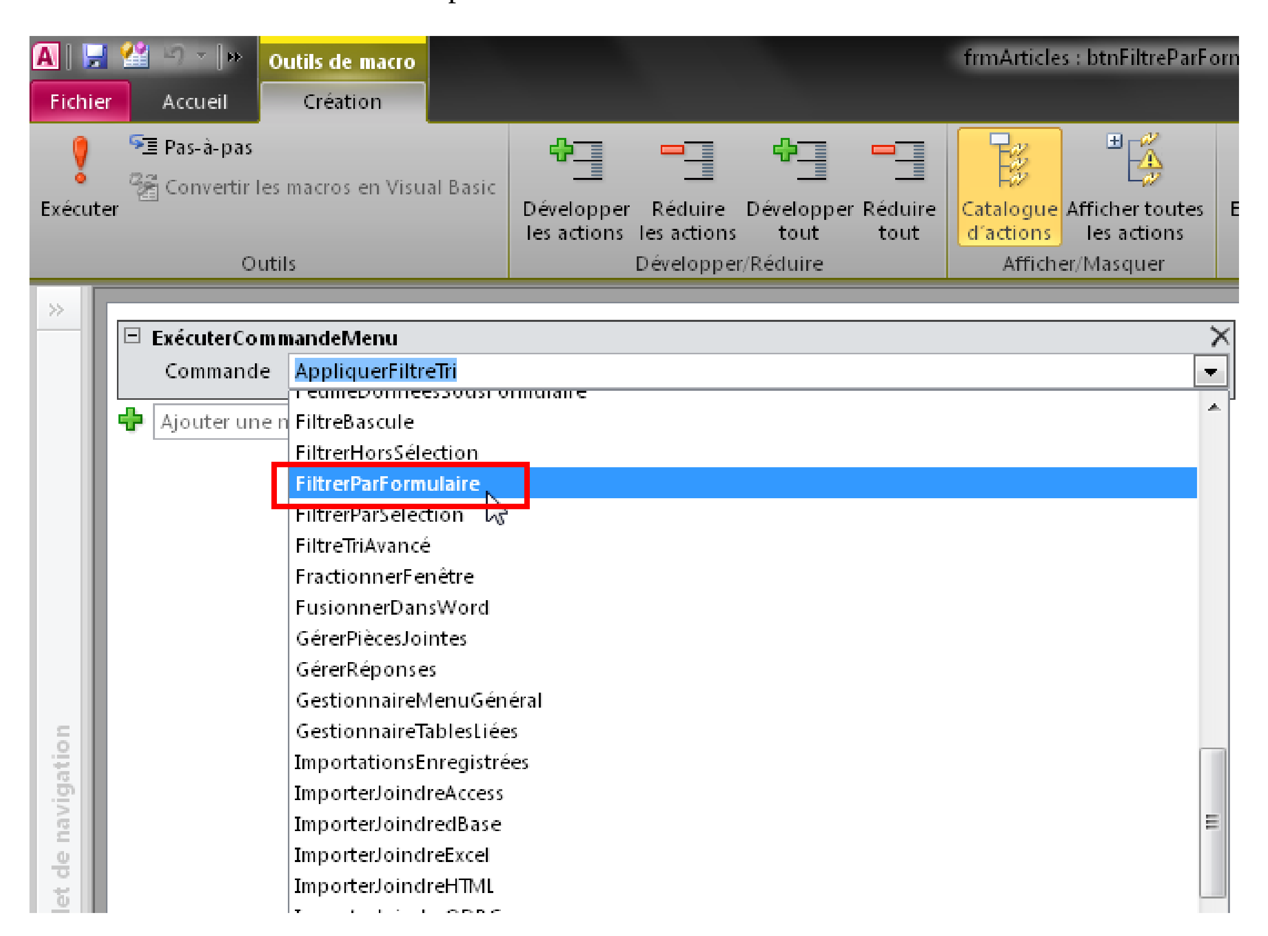
Ensuite, nous cliquons sur *Terminer*. Cependant le bouton ne fera pas un filtre par fomulaire... il faut alors aller modifier l'action de l'événement *Sur clic* associée:



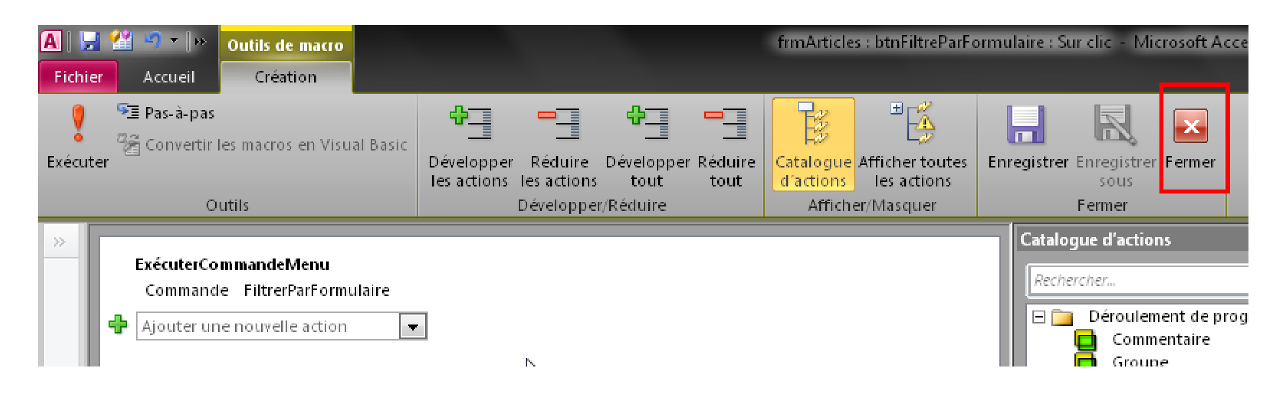
Nous arrivons alors (du moins depuis la version 2010 de Microsoft Access):



Nous ouvront la liste déroulante pour aller chercher l'action:



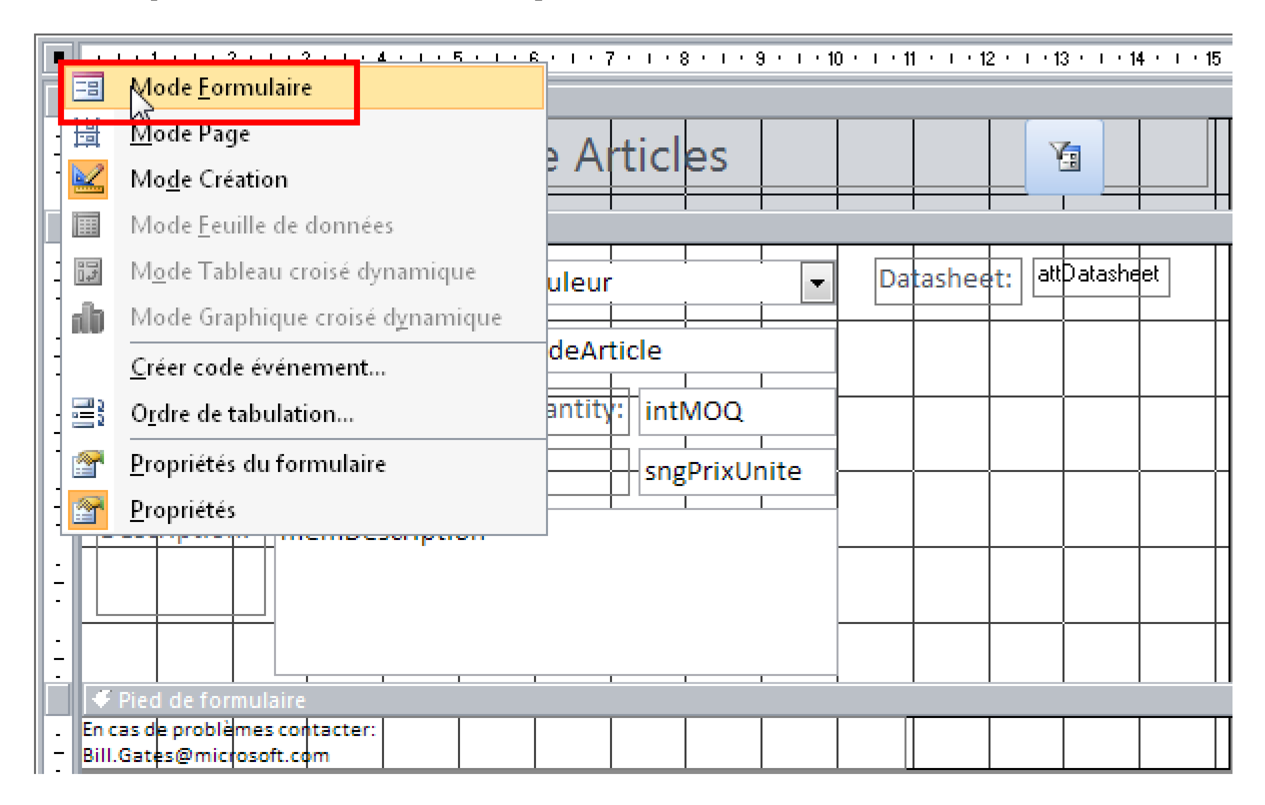
Une fois ceci fait, nous cliquons sur le bouton *Fermer*:



et à la question qui vient:



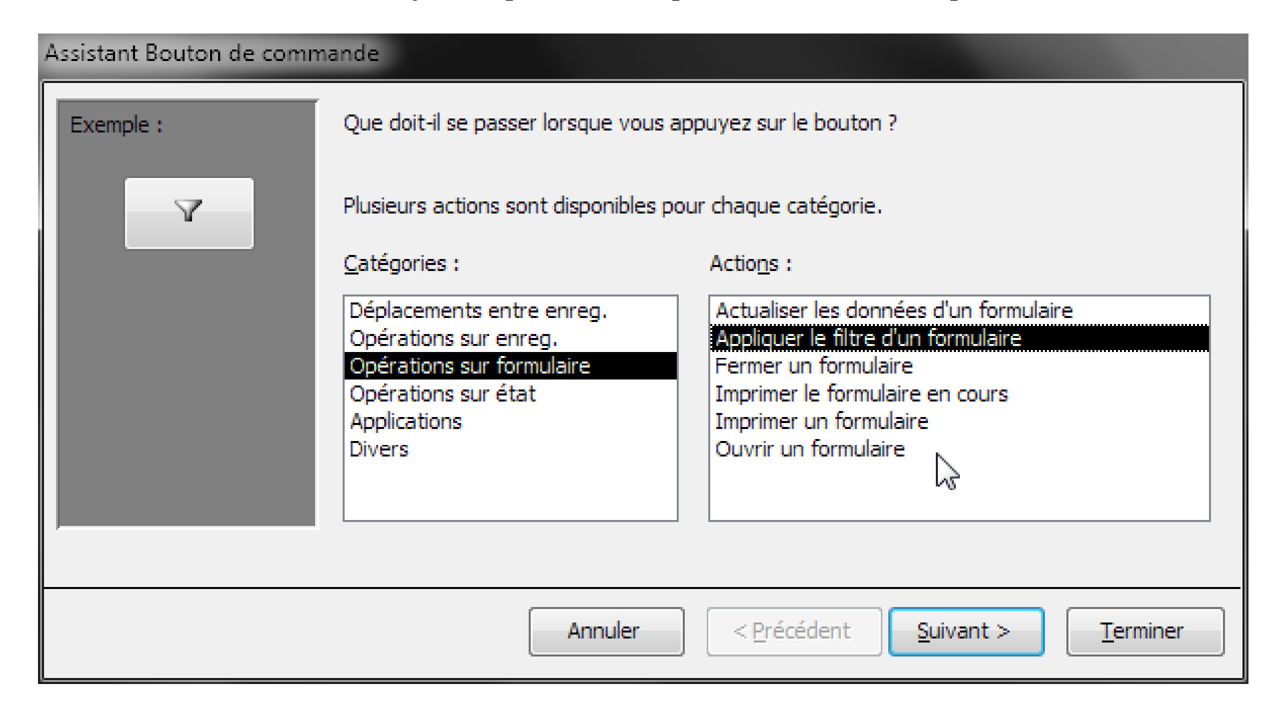
Nous cliquons sur *Oui*. Ensuite nous repassons le formulaire en mode *Formulaire*:



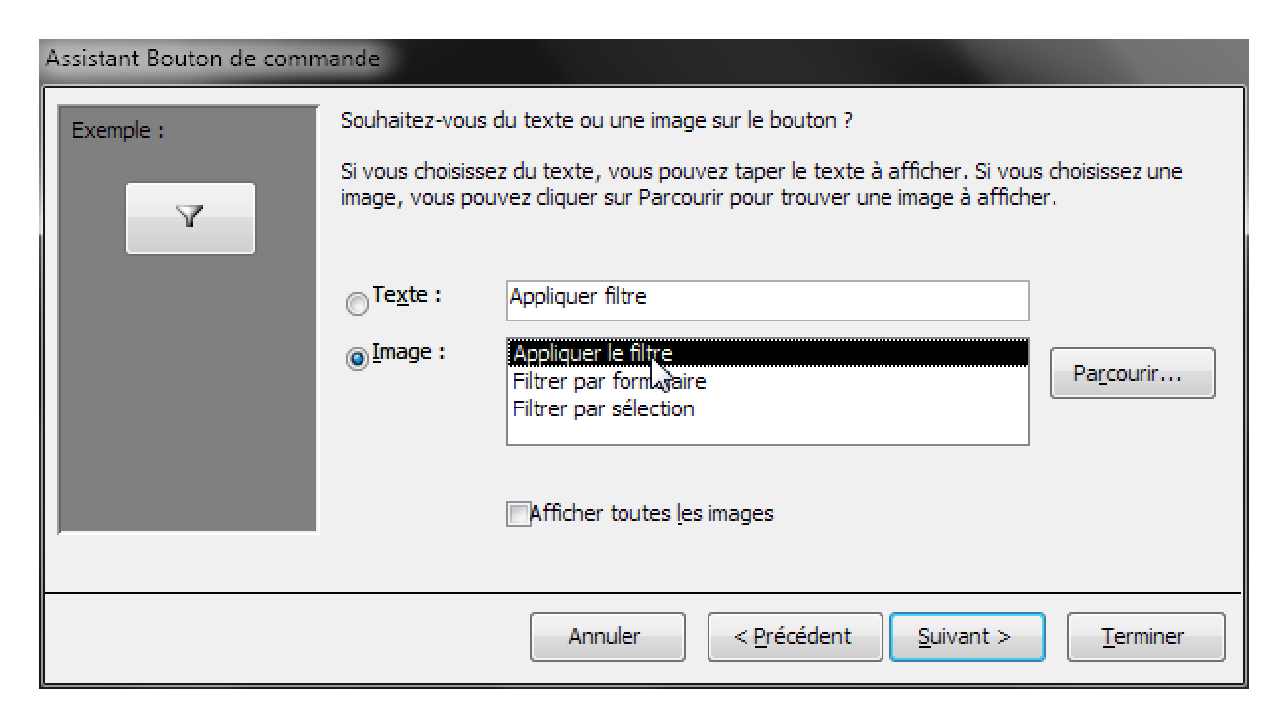
Ce qui donne après que nous ayons cliqué sur ce nouveau bouton:



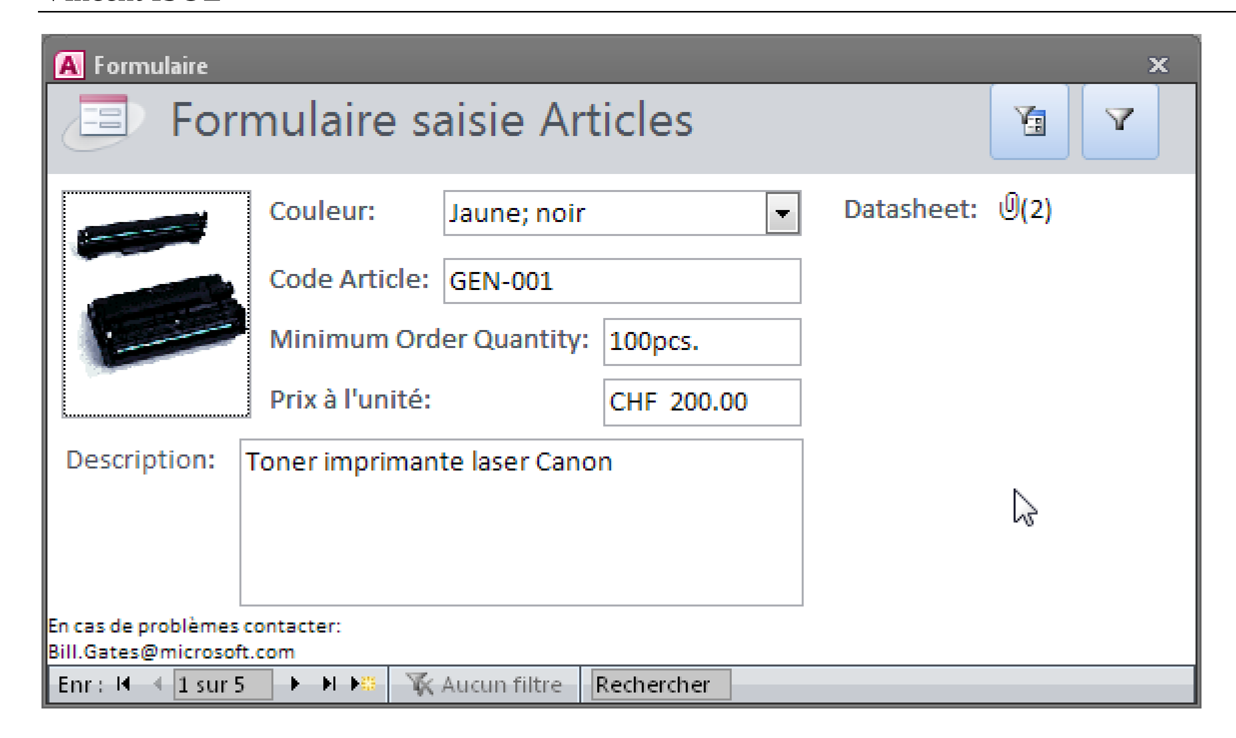
Donc cela marche. Maintenant il nous faut un bouton pour appliquer les critères de l'utilisateur. Nous recommençons la procédure depuis le début mais en prenant cette fois-ci:



et:



Une fois que nous arrivons à la fin nous obtenons:



Le problème est qu'activer le filtre par formulaire **désactive les boutons présents sur le formulaire...** (voir les remarques en rouge au début de cette section). Donc nous sommes coincés... Il semblerait que seule possibilité soit de créer un ruban contextuel ou une barre d'outils contextuel mais cela sort du cadre de ce chapitre.

C'est pourquoi nous allons voir, dès maintenant, comment manipuler correctement un outil aussi puissant et pratique que sont les filtres par requêtes.

6.5 Filtres requêtes

Créez un formulaire pour la table *tblSorties* enregistrez-le sous le même nom. Ensuite, créez deux filtres distincts pour les articles sortis durant l'année 1997 et 1996 que vous enregistrerez respectivement dans deux requêtes différentes (*qryArtSort1997* et *qryArtSort1996*).

Passez votre formulaire *frmSorties* en mode création et ajoutez deux boutons qui exécuteront respectivement ces deux "filtres-requêtes".

Ce qui est bien sûr regrettable, c'est que l'utilisateur ne peut choisir facilement les dates voulues. Nous verrons comment pallier ce problème en utilisant toute la puissance des objets de type "Requêtes" dès que nous les étudierons en détail.

Cependant, afin que vous ne soyez pas frustré, regardez ce que va faire votre formateur pour ajouter de l'interactivité avec l'utilisateur et vous verrez que cela est "simple comme bonjour".

6.6 Outil recherche

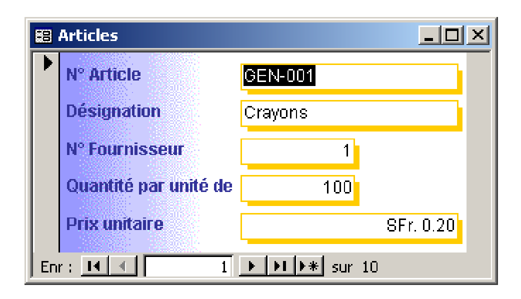
Il est très fréquent que l'on doive chercher des données dans une base. Pour cela, il faut que l'utilisateur ait les outils nécessaires à sa disposition. Et il en a quatre:

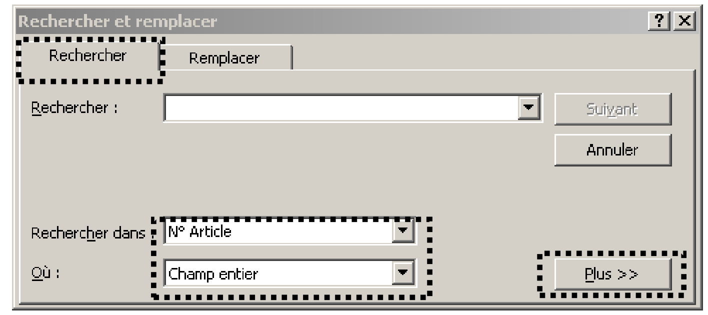
1. l'outil de recherche (recherche et remplacer) inclut dans MS Access et disponible par l'accès au menu *Edition* ou par la barre d'outils mais inconnu des utilisateurs lambda

2. l'outil de recherche (recherche et remplacer) inclus dans MS Access mais dont vous avez créé un *raccourci* sur le formulaire à l'aide de l'assistant de boutons.

3. l'outil de recherche par listes (combo box ou list box) inclus dans MS Access mais dont vous avez créé l'accès en utilisant l'assistant correspondant de la boîte à outils contrôle (voir plus loin)

4. le développement d'un formulaire spécial dédié à la recherche derrière lequel se cache du code VBA (si la recherche est élaborée…)





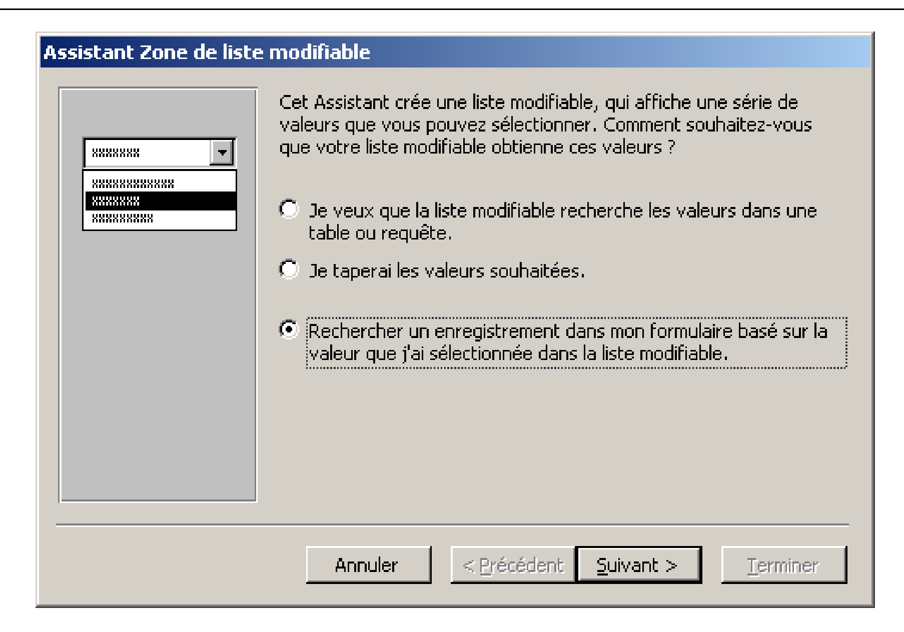
Voyons comment marchent les deux premières options (la troisième est vue beaucoup plus loin en cours).

6.7 ComboBox de recherche

Nous allons maintenant créer un petit (tout petit) et très simple outil de recherche sur notre formulaire (qui peut se substituer au Rechercher/Remplacer).

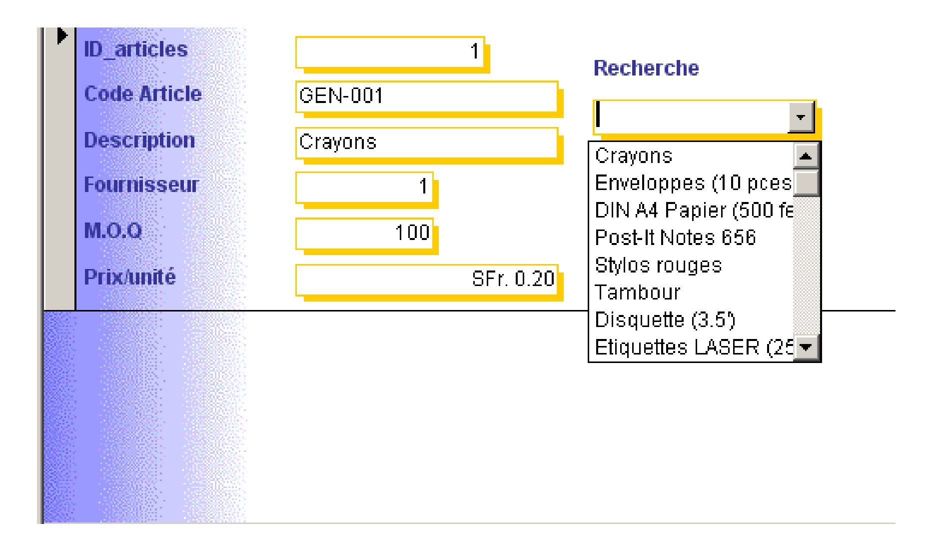
Pour cela, placez votre formulaire en "Mode Création" et utilisez la palette d'outils pour ajouter une ComboBox (suivez la démarche de votre formateur et prenez des notes car c'est une fonction importante).

Une fois la ComboBox placée (à un endroit qui au niveau de la surface le permet), la fenêtre ci-dessous doit apparaître:



Choisissez la troisième option, cliquez sur "Suivant" et choisissez l'élément de votre formulaire qui va servir de critère de recherche (le champ *Désignation* serait un bon choix..).

Et voilà ce dont à quoi le résultat devrait ressembler:



Il suffit maintenant de choisir un champ dans la liste pour accéder directement au premier enregistrement correspondant.

Remarques:

R1. Vous pouvez ajouter autant de "listes de recherche" que vous désirez

R2. Plusieurs listes de recherche ne cumulent pas les critères (ET logique)

R3. Cette fonctionnalité marche également en mode "tabulaire" sans cacher toutefois les enregistrements qui ne correspondent pas au critère (MS Access se positionne seulement sur l'enregistrement intéressé).

R4. Pour appliquer un filtre plus complexe, préenregistré, il faudra d'abord que nous étudions comment créer des requêtes paramétrées simples et ensuite comment créer une macro utilisant l'option "Appliquer un filtre".

Ce dernier point consiste en un joli exercice de style qui n'est pas vraiment prévu dans la formation faute de temps. Mais cependant, si une majorité de participants souhaitent voir comme cela marche, il faudra qu'ils s'accrochent (dans le cadre d'un cours de base) à leurs stylos et qu'ils soient prêts à sacrifier 20 à 40 minutes sur le programme prévu. Sinon, si le participant au cours souhaite y revenir lors d'une formation avancée il faudra qu'il prenne garde à rappeler le formateur d'avoir l'obligeance de montrer cette fonctionnalité (eh oui ! j'oublie parfois… tellement il y de choses à montrer…).

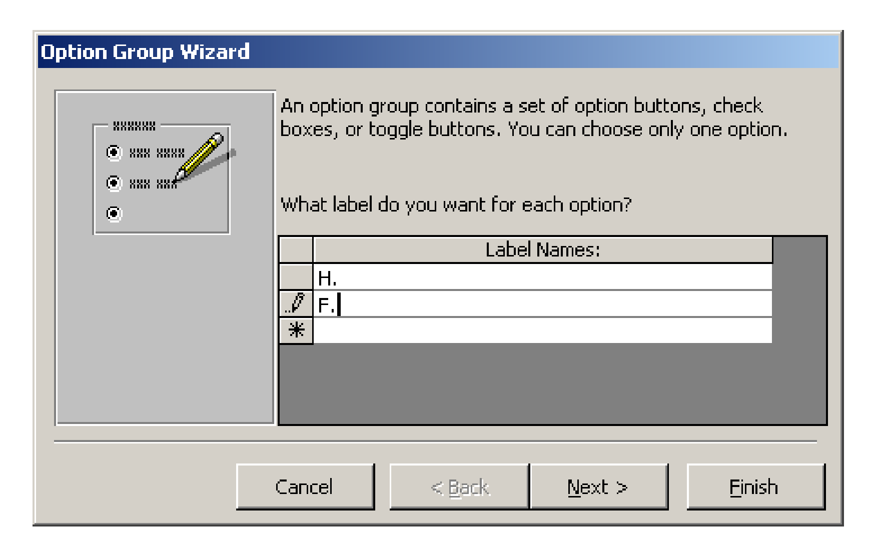
6.8 Groupes d'options

Et les boutons d'option ? Qu'en est-il ? Eh bien c'est simple, créons dans la table vendeurs un nouveau champ nommé *IntGenre* de type *Integer*.

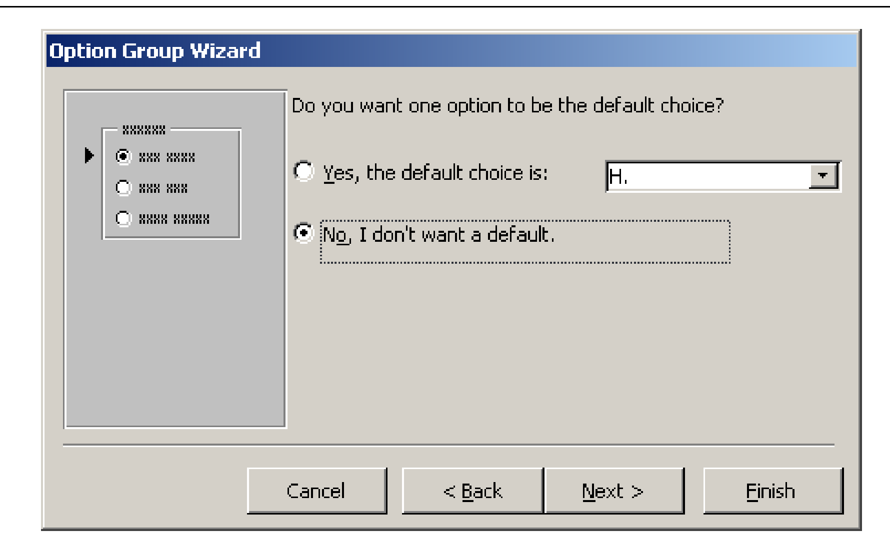
Ensuite, dans le formulaire, relatif à la saisie de nouveaux vendeurs, passez en mode création et sélectionnez la boîte de regroupement dans la barre d'outils:



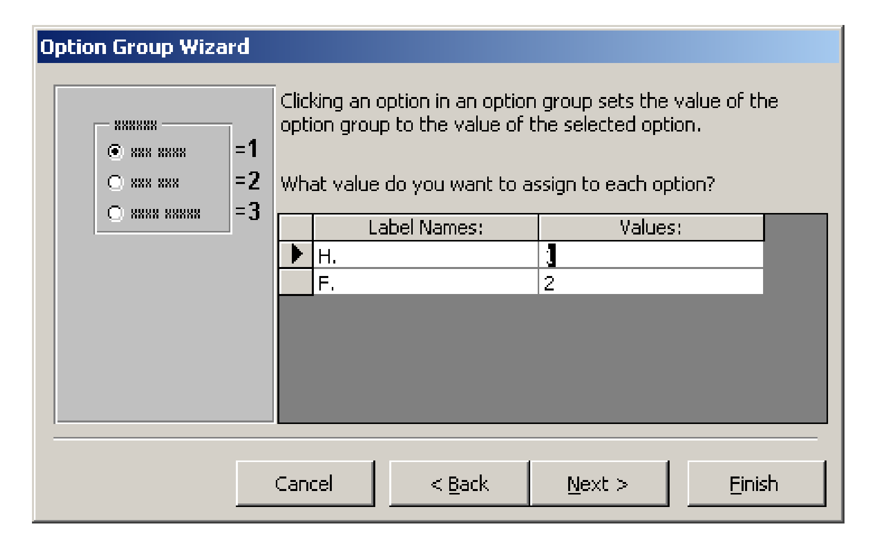
Définissez une telle zone à un endroit voulu du formulaire. Ceci fait, la boîte de dialogue suivant apparaît:



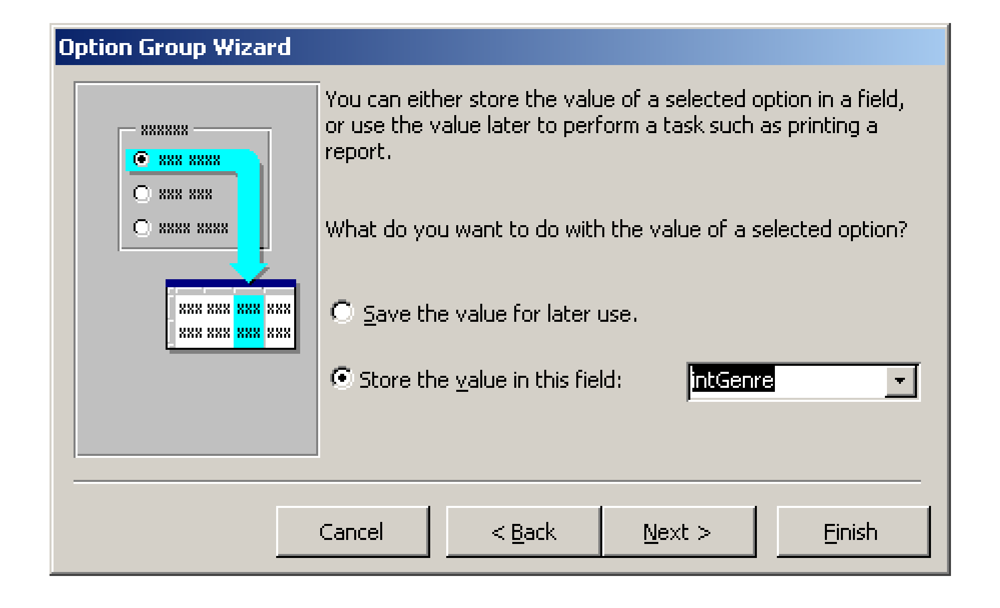
Précisez que vous ne voulez aucune valeur par défaut:



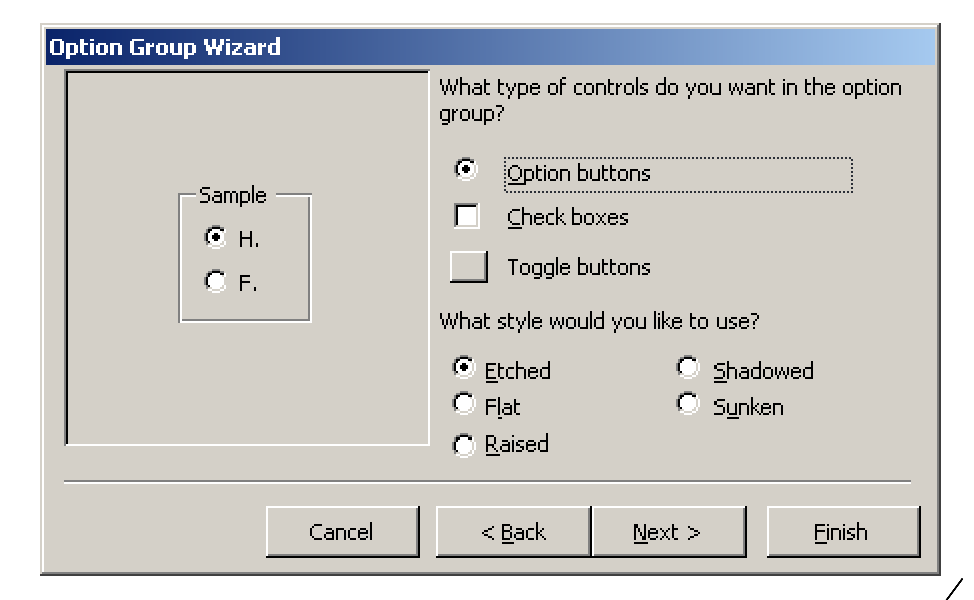
Ensuite vous arrivez à la fenêtre suivante (où il est inutile dans cet exemple de changer quoi que ce soit):



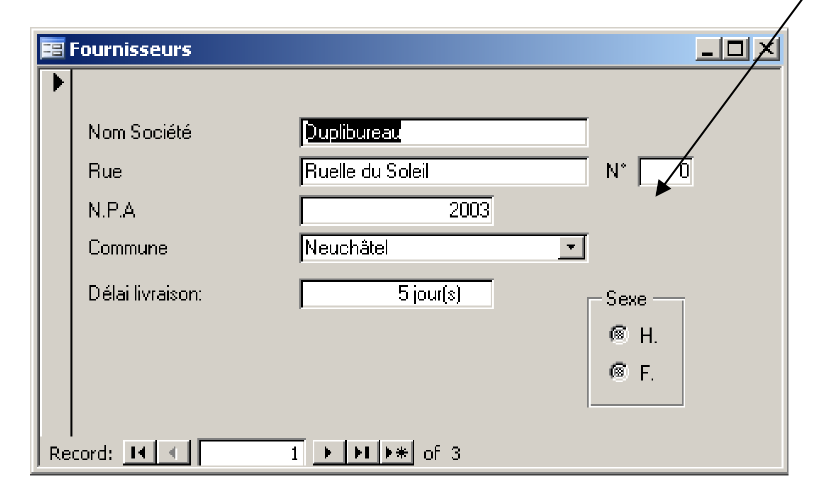
Ensuite, il faut bien évidemment choisir le champ dans lequel va être enregistrée cette valeur, en l'occurrence *intGenre:*



Ensuite, vous avez plein de choix intéressants et pertinents pour l'aspect visuel::



Il suffit ensuite de donner une légende au groupe et vous obtenez finalement un formulaire du type:

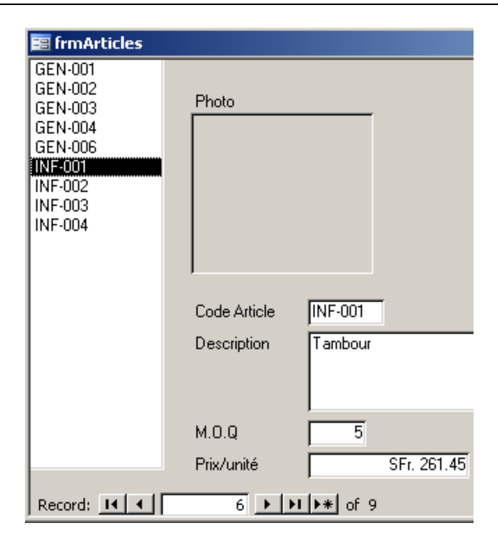


6.9 Champs calculés

Imaginons le scénario suivant: un collègue vous demande de pouvoir visualiser dans la table articles le prix de 1 MOQ (en d'autres termes: les prix à l'unité multiplié par la quantité minimale). Comment allez-vous procéder pour afficher cette information dans le formulaire qui affichera les informations relatives à la base MS Access?

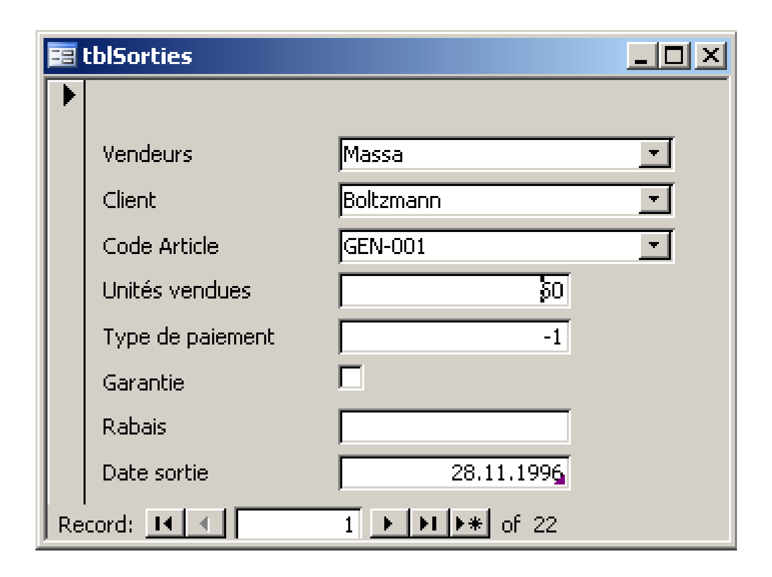
6.10 ListBox de recherche

Créez un formulaire pour la saisie de nouveaux articles et insérez une liste de recherche de manière à avoir la chose suivant à l'écran (c'est plus courant que la liste déroulante):



6.11 Fonction DSum (SomDom en FR)

Considérons le formulaire (pas fini!) des sorties des articles ci-dessous:



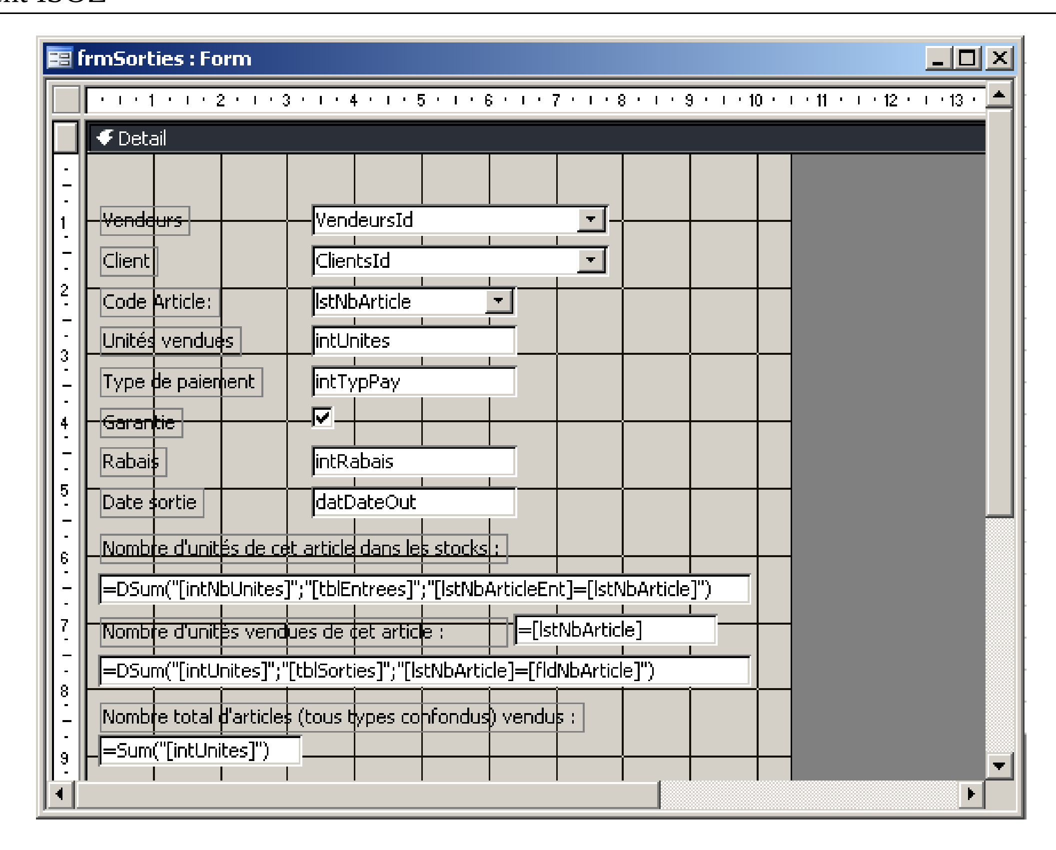
Les questions sont les suivantes:

1. Comment afficher toute la quantité du nombre d'unités vendue de tous les types d'articles

2. Comment afficher dans ce formulaire pour un article donné le nombre total d'unités vendues ?

3. Idem que (2) mais le nombre restant ?

Pour les autres, l'idée va consister à utiliser la fonction *DSum* (en anglais) et *Sum* de la manière suivant dans des champs que vous aurez créé (vous allez voir que la manière dont nous avons nommé les champs peut porter à confusion d'où l'importance d'une stratégie de nommage):



et il est bien évidemment possible d'utiliser d'autres fonctions que *Sum* pour faire des statistiques de la même table que celle sur laquelle est principalement basé le formulaire.

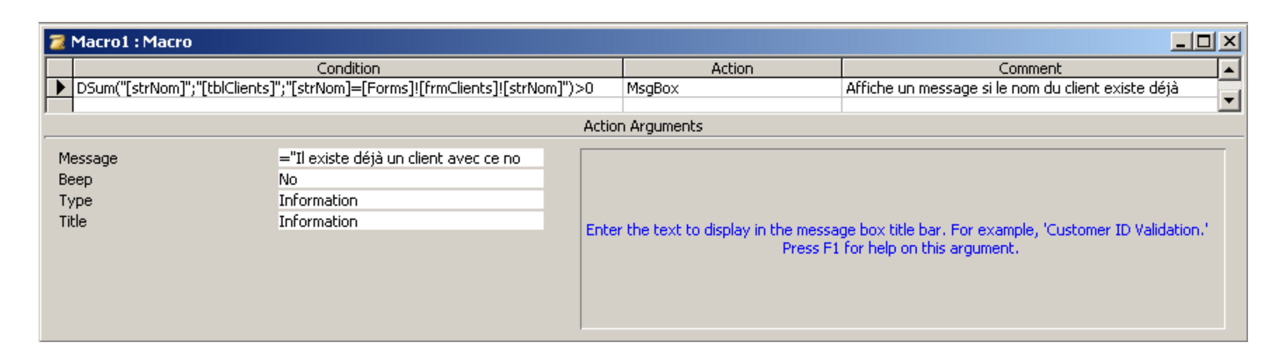
Remarque: Si pour le critère de la fonction DSum vous avez deux champs qui ont le même nom, spécifiez en suffixe la provenance (nom de la table ou de la requête entre crochets).

6.12 Fonction DCount (CpteDom en FR)

La fonction *DCount* a une syntaxe en tout point similaire à la fonction *DSum*. Elle compte simplement le nombre d'enregistrements au lieu de sommer leur valeur.

Elle peut être très pratique dans le cadre des macros. Effectivement, il est relativement aisé dans le formulaire de saisie des nouveaux clients ou vendeurs de faire en sorte que lorsque l'utilisateur quitte le champ *strNom*, la macro contrôle s'il existe déjà un vendeur du même nom (bon on peut faire pareil avec les index multiples mais l'idée peut être développée dans d'autres cas d'applications).

Un exemple qui peut être fait avec votre formateur dans le cadre du formulaire de saisie de nouveaux clients:



6.13 Fonction IIf (formulaire)

Nous allons voir maintenant l'équivalent de la fonction SI connue dans MS Excel mais en version MS Access. Pour cela, nous allons créer un champ dans le formulaire *frmArticles* qui affiche "Trop cher" si le prix à l'unité est supérieur à 100.- ou "OK" lors que le prix en est inférieur.

Remarque: la fonction SI dans MS Access est nommée *VraiFaux* ou *IIF* en anglais:



6.14 Fonctions d'environnement

Signalons encore deux fonctions très utiles dans les formulaires, requêtes ou rapports (cette fonction ne marche pas dans les tables!):

=Environ("username")

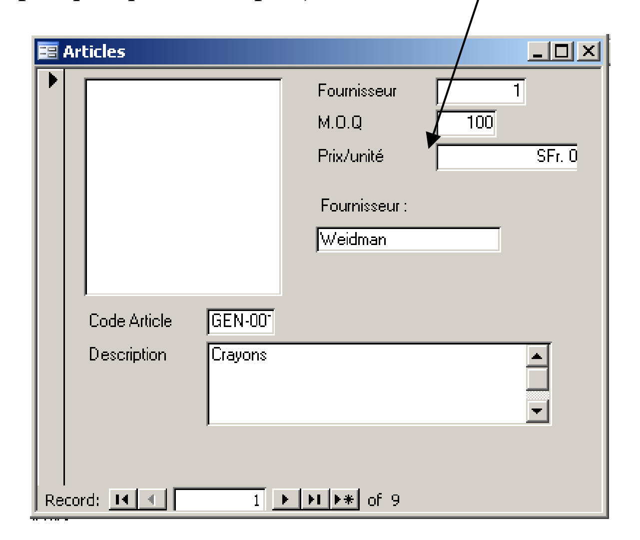
qui renvoie le nom de l'utilisateur de la session. Ou si elle ne fonctionne pas:

=CurrentUser( )

6.15 Fonction DLookUp (RechDom en FR)

Petit exercice de style… et très demandé (ainsi que découverte d'une des fonctions les plus puissantes de MS Access):

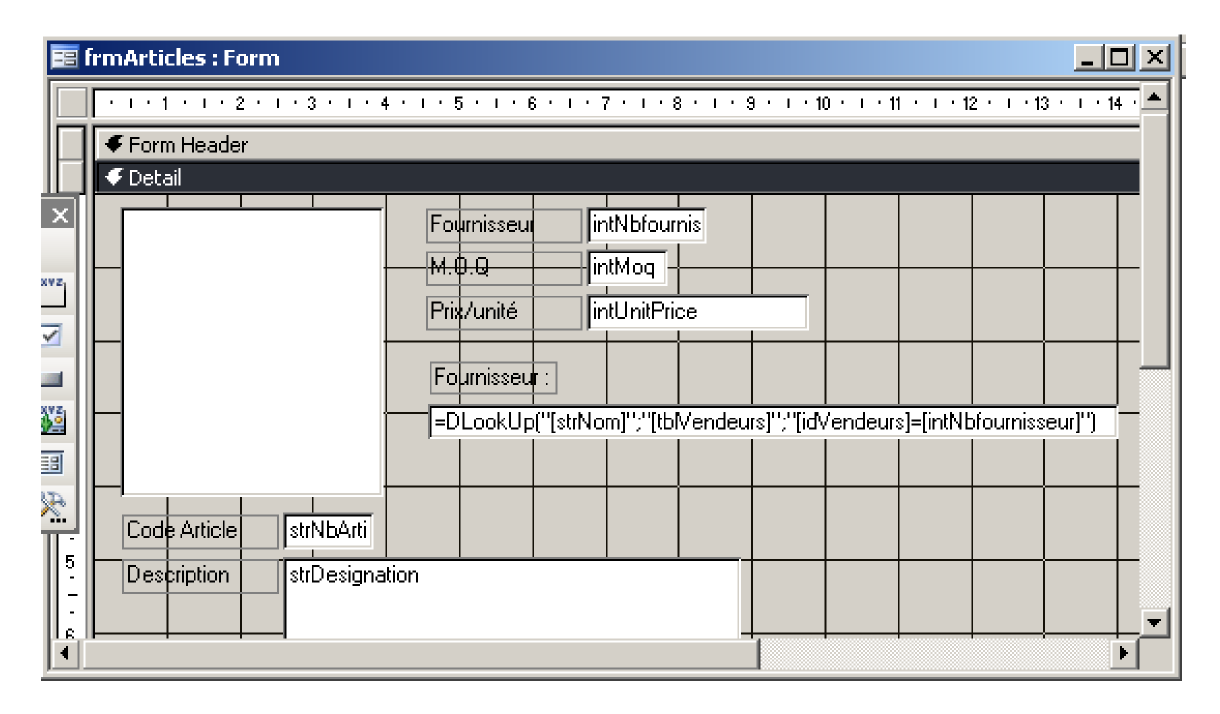
Comment afficher (par exemple) sur le formulaire de saisie d'articles, le plus simplement et esthétiquement possible, pour chaque article le nom du fournisseur (et non pas l'id !) correspondant à un article donné tel que présenté ci-dessous (et pas autrement mis à part l'emplacement du champ à quelque chose prêt):



Essayez tout d'abord avec l'assistant de formulaire de trouver le résultat…

Remarque: si vous avez déjà le nom du fournisseur (car cela dépend de votre choix antérieur lors du cours de base où nous faisons exprès de créer des erreurs !) faites en sorte d'avoir le délai de livraison qui s'affiche sur le formulaire en fonction du nom du fournisseur.

La solution (il faut bien que vous l'ayez quelque part mais ne "trichez" pas en passant tout de suite à celle-ci sinon vous ne verrez pas l'intérêt de l'exercice) consiste à utiliser la très fameuse fonction DLOOKUP (RechDom en français) e Access et dans un nouveau champ, d'écrire la fonction suivante:



Faites de sortes que toutes les "champs calculés" soient verrouillées et inactifs afin que l'utilisateur ne puisse pas cliquer dedans (c'est une opération qu'il faut aussi savoir faire dans les formulaires ressortant de requêtes comme nous le verrons plus tard).

En tant qu'exercice, refaites de même que précédemment mais avec le délai de livraison (c'est plus pertinent) ainsi qu'avec le canton de localisation du fournisseur et le pays.

Petite remarque ! Rappelez-vous qu'au cours de base nous avions fait exprès de mettre les clés primaires au mauvais endroit dans les tables *tblPays* et *tblCantons*. Nous vous demandons dès lors la chose suivante afin de vous exercer à nouveau avec les concepts de clés primaires:

1. Pour le canton, ne changez rien
2. Pour le pays, corrigez l'emplacement de la clé primaire

Quelles sont les différences dès lors au niveau de l'utilisation de la fonction DLookUp ?

6.15.1 Équivalent de DLookUp en cas d'utilisation de GUID

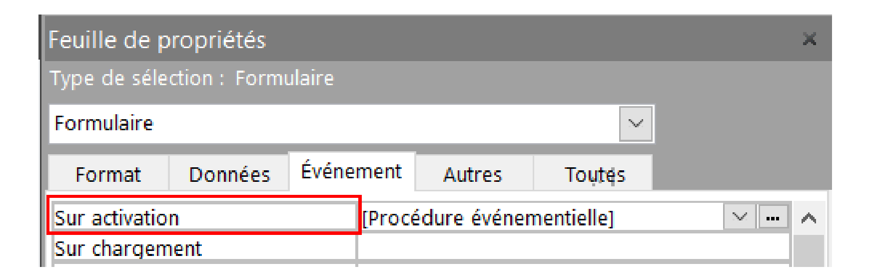
Comme nous l'avons précisé au début de ce livre, DLookUp et un certain nombre de fonctions ne vont pas marcher avec les GUID. Alors comment réussi ce que nous avons fait ci-dessous dans le cas d'utilisation de GUID en lieu et place de clés primainres de type Entier Long?

Eh bien d'abord vous oubliez de mettre une formule directe dans un champ *lblFournisseur* où nous avions mis la formule ci-dessous. Vous mettez le code VBA suivant:

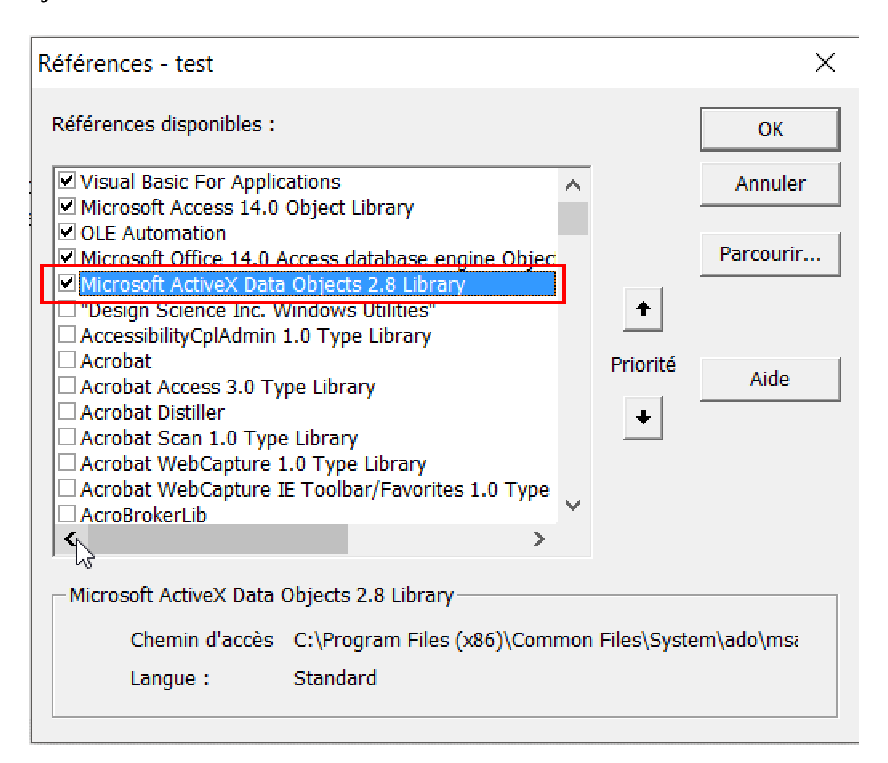
```
   On Error GoTo ErrorHandler
   Dim MySet As Recordset
   Set MySet = CurrentDb.OpenRecordset("SELECT strNom FROM tblVendeurs WHERE
idVendeurs=" & StringFromGUID(Me!intNbfournisseur))
   Me!lblFournisseur = MySet!strNom
   MySet.Close
   Exit Sub

ErrorHandler:
   Me! lblFournisseur = ""
```

dans l'événement *Sur Activation* du formulaire (ce qui aura pour effet d'être exécuté à chaque changement d'enregistrement aussi!!!!):



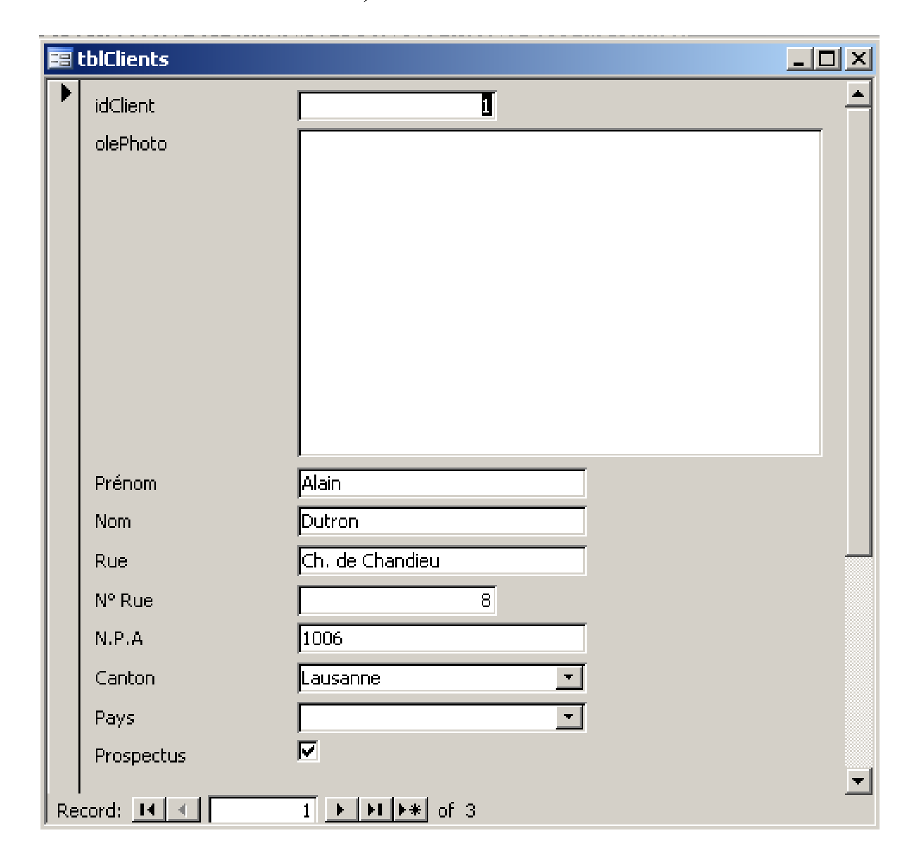
Sans oublier d'ajouter la référence suivante:



6.16 Onglets

Lorsque vous avez trop de données à représenter dans vos formulaires, vous avez la possibilité dans les formulaires de MS Access d'ajouter des zones d'onglets pour ranger vos champs par défaut ou vos champs supplémentaires.

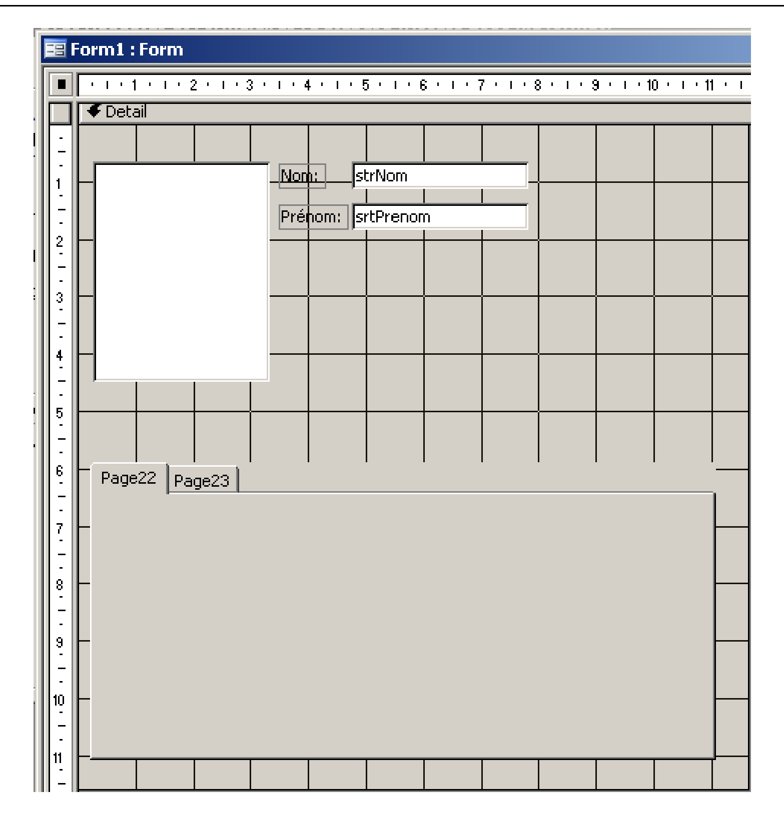
Créez un formulaire automatique de la table *tblClients* par exemple (elle contient peu d'informations mais nous ferons avec…):



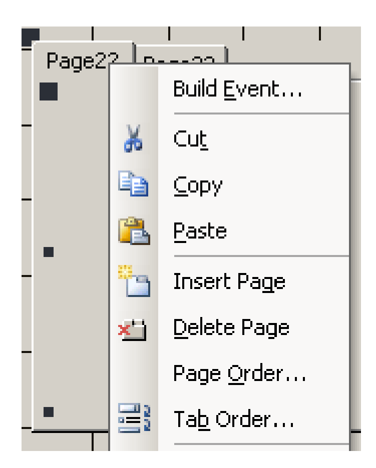
et passez en mode création. Sur la boîte à outils vous avez le bouton suivant:



Dans votre formulaire faites un peu de place (en supprimant les champs que vous voudrez dans les onglets !) pour les futurs onglets et insérez en un en le dessinant (comme vous le feriez dans MS Word):

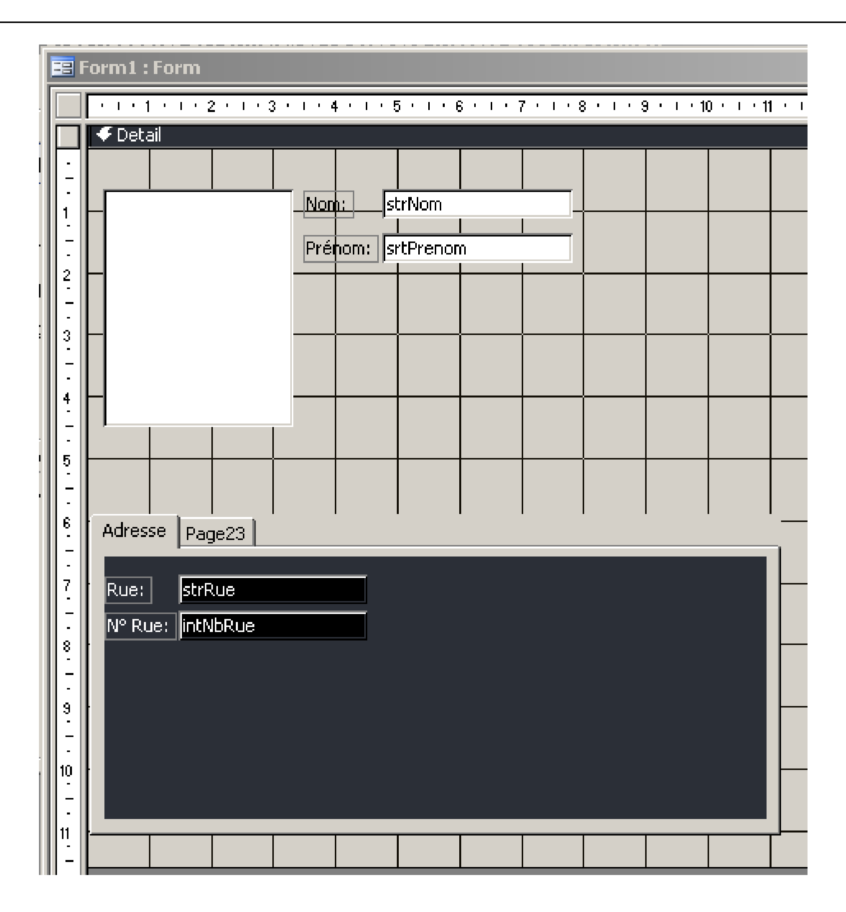


Dans un premier temps, pour renommer, supprimer ou ajouter des onglets, il suffit de faire un clic droit sur le nom de ceux-ci tel que:



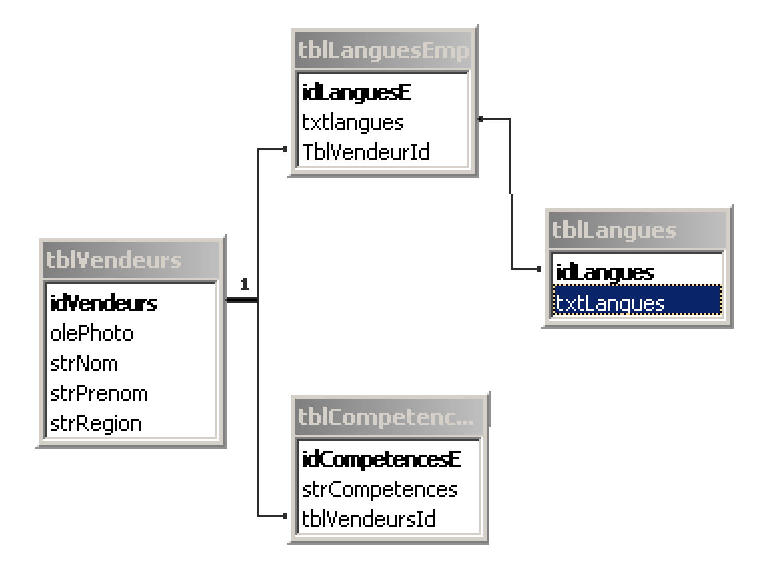
aussi simple que ça…

Pour mettre les champs par défaut de la table dans les onglets, il suffit de les y glisser depuis la fenêtre des champs (voir page suivante):



6.17 Valeurs par défaut et filtres

Si cela n'est pas déjà fait, faites en sorte que l'employer puisse spécifier pour un vendeur les langues maitrisées ainsi que la liste des compétences tel que:



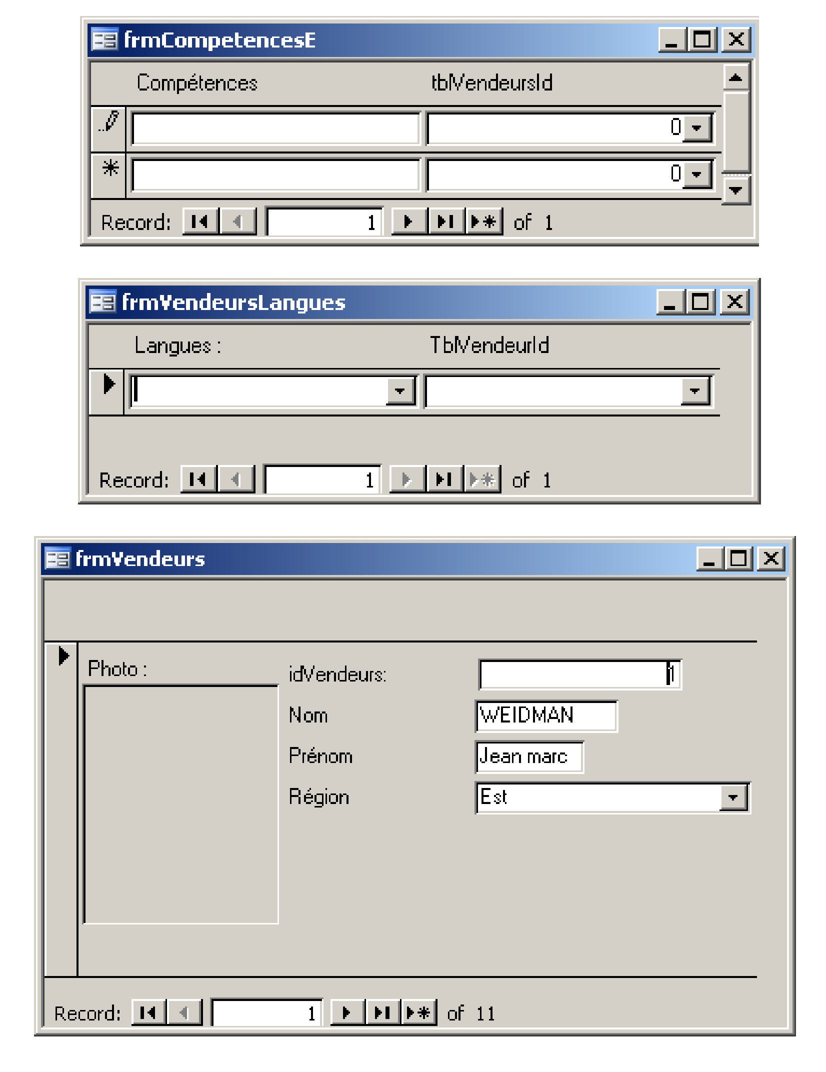
Comment faire en sorte que dans le formulaire vendeurs, nous puissions saisir indépendamment les langues et les compétences d'un vendeur donné ?

1. Créer un formulaire en mode tabulaire de la table *tblCompetencesV* (avec les champs *strCompetences* et *tblVendeursId*) et l'enregistrer sous le nom *frmComptencesV*

2. Créer un formulaire en mode tabulaire de la table *tblLanguesV* (avec les champs *strLangues* et *tblVendeursId*) et l'enregistrer sous le nom *frmLanguesV*

3. Créer un formulaire en mode simple (avec le champ *IdVendeurs*) de la table *tblVendeurs*

Ainsi (pour l'esthétique nous vous laissons faire…):



Passez maintenant le formulaire *frmVendeurs* en mode création et avec l'assistant de création de boutons, créez un bouton pour ouvrir le formulaire de langues *frmLanguesV* .

Ensuite, dans le formulaire *frmLanguesV* dans le champ *tblVendeurId* écrivez la formule:

=[Forms]![frmVendeurs]![idVendeurs]

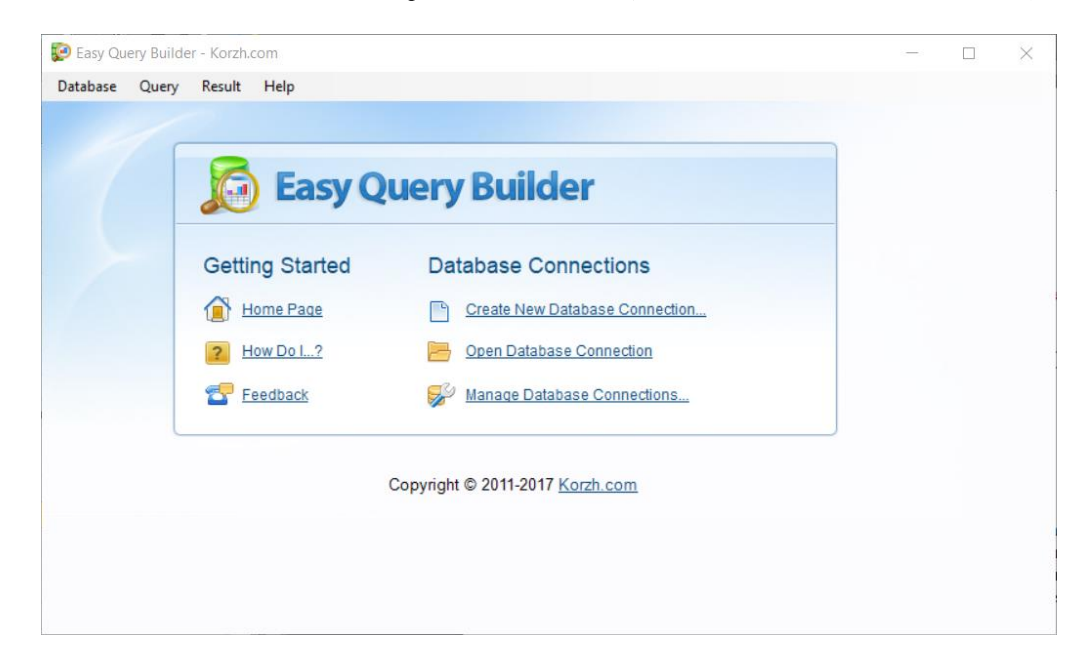
et de même pour les compétences. Ensuite, il vous suffit de masquer à l'utilisateur les champs inutiles pour lui et voilà que le système fonctionne maintenant parfaitement.

7 Requêtes (simples)

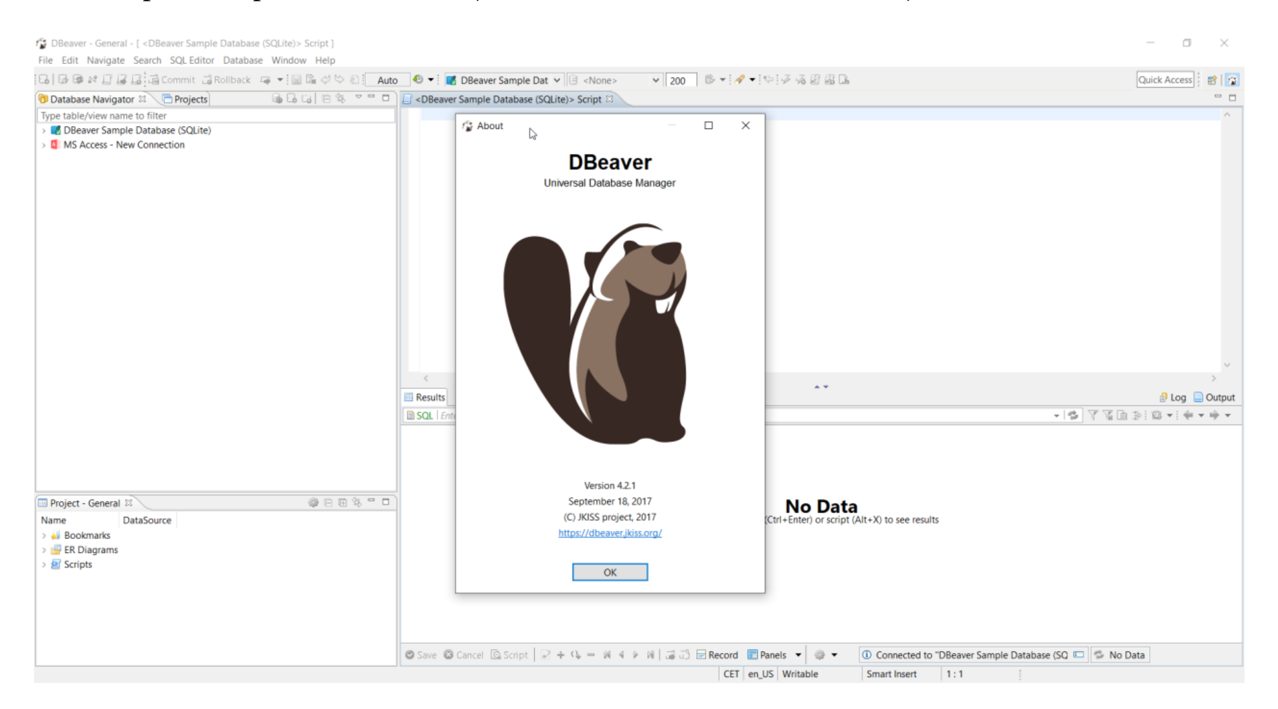
Attention! Enregistrez toujours une requête avant de l'exécuter et veillez à respecter la nomenclature de L-R.

Microsoft Access dispose d'un des QBE (Query By Example) les plus souples et les plus puissants du marché. Il est aussi facilement abordable par un débutant capable de créer tous types de requêtes d'actions, d'analyses croisées ou d'union.

Cependant il existe des alternatives gratuites comme (Microsoft Windows seulement):



et sinon pour les puristes du SQL (Microsoft Windows+Mac+Linux):



Il est aussi très simple de créer des requêtes imbriquées en enregistrant la première puis en l'appelant par son nom, dans la requête parent. Non seulement, cette fonctionnalité permet d'optimiser des requêtes complexes comme on peut le faire en SQL (Structured Query Language), mais elle clarifie la vue d'ensemble et permet une mise au point par étapes: tester d'abord le niveau le plus bas, puis remonter…

Sans parler des assistants qui font gagner un temps précieux: recherche de doublons

Ce QBE est tellement rapide et simple que, si j'ai l'obligation, dans un programme VB, C#, ASP, PHP ou autre de manipuler du code SQL, je n'hésite jamais à:

1. Ouvrir une base MS Access
2. Attacher les tables nécessaires
3. Créer une requête afin d'obtenir le résultat désiré
4. Afficher les données pour vérifier que les résultats correspondant aux attentes
5. Copier le code SQL dans l'application VB, C#, ASP, PHP ou même SQL Server
6. Remplacer les valeurs paramètres par des noms de variables

Différences entre filtres et requêtes:

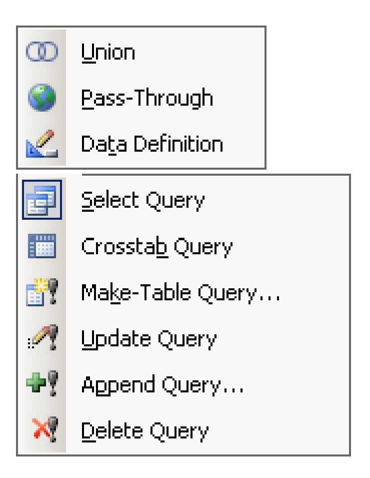
	Filtres	**Requêtes**
Sélection des données	Une table est filtrée par ligne pour une table unique	Une requête peut être créée pour plusieurs tables d'un coup
Enregistrement	Le filtre est enregistré dans la table et ne peut être consulté qu'en ayant la table à l'écran.	La requête est liée à une table-requête générée automatiquement à chaque exécution de la requête et peut être visualisée dans un formulaire.
Utilisation	Pour la sélection rapide et simple d'enregistrements en mode création.	Pour l'affichage et la saisie d'enregistrements en fonction de plusieurs critères dans des formulaires liés à la table-requête.

Les requêtes sont plutôt utilisées pour que l'utilisateur puisse modifier une donnée ou visualiser les enregistrements d'une table à partir du mode de création ou d'un formulaire. En aucun cas, il ne s'agit (communément en tout cas) d'un outil qui doit amener à faire de la saisie.

Il existe plusieurs types de requêtes différenciées sous MS Access (votre formateur va vous dire de quoi il s'agit et comment utiliser l'aide dans le cadre de MS Office 2003 et du code SQL):

1. Requêtes de sélection
2. Requêtes d'ajout

3. Requêtes de synthèse
4. Requêtes croisées
5. Requêtes de création
6. Requêtes de suppression
7. Requêtes de mise à jour
8. Requête de correspondance
9. Requête de non-correspondance
10. Requêtes d'union (à coder en SQL)
11. Requêtes de passage (pour exécution sur SQL Server sans passer par Jet)
12. Requêtes de définition (à coder en SQL)



Dans une "requête de synthèse" (dans le cas de données numériques) on peut calculer des totaux, moyennes et autres (il y aura un exercice là-dessus).

La case à cocher dans le mode création des requêtes permet de se servir d'un champ comme critère de la requête tout en l'obligeant à ne pas s'afficher dans la table résultant de la compilation de la requête (et dans le formulaire rattaché).

Pour lancer une requête, il suffit de cliquer sur le bouton:



Dans les cellules de la ligne de "Critères" il est possible de définir des inégalités qui établiront si un champ d'un enregistrement sera affiché dans la table compilée ou non. Par exemple, si nous souhaitons afficher un enregistrement uniquement s'il satisfait à une certaine inégalité donnée par des opérations mathématiques en relation à d'autres champs du même enregistrement. Ainsi dans la ligne de critères on écrira quelque chose du genre:

[Nom d'un premier champ]/[Nom d'un deuxième champ][Nom d'un troisième champ]>100*

Si cette inégalité est satisfaite alors le champ correspondant à la colonne de critère où l'on a saisi cette formule sera affiché.

On peut également créer dans la table compilée, une nouvelle colonne avec un "champ calculé". Ainsi, dans la ligne Champ on écrira:

NomNewColumn:[Champ existant]/Nombre voulu

Cela aura pour effet de créer lors de la compilation de la requête, une nouvelle colonne nommée "NomNewColumn" avec les résultats du calcul saisi.

Remarques:

R1. Nous n'allons voir ici que les requêtes "simples". Les autres seront vues dans le chapitre nommé "Requête et jointures" (voir page 216).

R2. Le lecteur remarquera lors de l'utilisation des assistants d'Access qu'il est possible d'utiliser une requête dans une requête et de la mélanger avec des tables pour faire des analyses pertinentes.

7.1 Optimisation des requêtes

Signalons pour la culture générale qu'il y a plusieurs manières d'optimiser la rapidité d'exécution des requêtes (quand on traite quelques millions de données cela devient utile!):

1. Augmenter la puissance des ordinateurs (logique…)

2. Effectuer des modifications de la base de registre MS Windows

3. Désactiver les messages d'avertissement de MS Access (ralentit considérablement)

Nous approfondirons un peu plus le sujet dans le chapitre 35.

7.2 Requête simple (de projection)

Définitions:

D1. Une "requête de sélection" est une requête qui produit une nouvelle table étant un sous-ensemble de la table originale selon des critères appliqués sur certains champs (l'opérateur d'algèbre relationnel correspondant est σ).

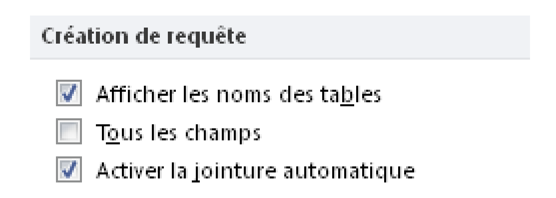
D2. Une "requête de projection" est une requête qui produit une nouvelle table qui est un sous-ensemble de la table originale selon un ou des champs sélectionnés (l'opérateur d'algèbre relationnel correspondant est π).

Avec l'assistant requêtes créez une "Requête Simple" (voir qu'il existe également une requête pour les doublons comme on en avait parlé auparavant) pour la table *tblArticles* ne contenant que les champs du numéro d'article et du prix unitaire. Enregistrez la requête sous le nom *qryPrixArticle* et créez-en un formulaire instantané. Fermez le formulaire instantané sans l'enregistrer.

Question: pouvons-nous rajouter de nouveaux articles à partir du formulaire de la requête. Argumentez votre réponse !

Remarque: Le raccourci clavier pour passer d'une vue de requête à l'autre est "Ctrl+Point" et "Ctrl+Virgule" pour avancer ou reculer dans les vues. Dans la partie Macros du présent ouvrage, on verra comment créer un unique raccourci pour exécuter les requêtes.

Avant de commencer à faire une requête quelconque, évitez de cocher l'option *Tous les champs* suivante dans la configuration du logiciel:



Effectivement, à partir du moment où vous cochez cette case, toutes les nouvelles requêtes que vous créérez vous afficheront TOUS les champs des tables présents dans la requête lors de son exécuation même ceux que vous n'auraz pas sélectionné comme devant s'afficher...

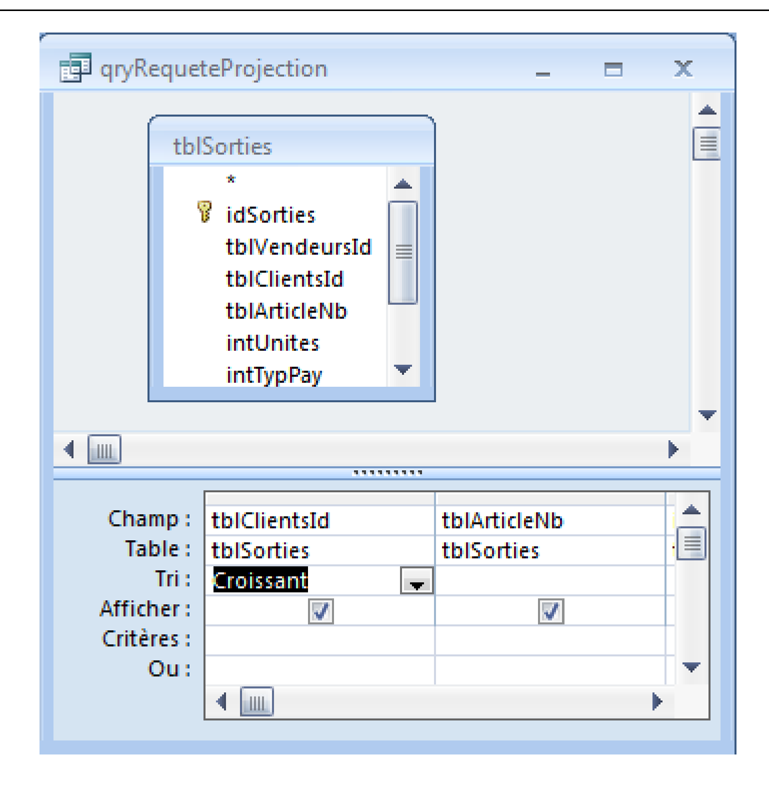
7.3 Tris dans les requêtes

D'abord, il faut savoir que MS Access applique les options de tris dans l'ordre dans lequel se trouvent les colonnes de la requête.

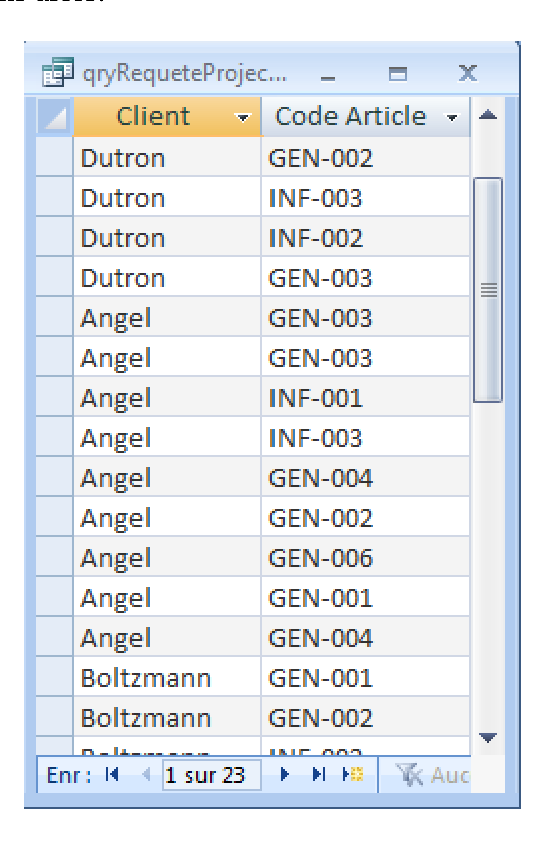
Ensuite, le but de cette petite section est de parler de deux problèmes courants des tris avec les requêtes.

Le premier problème ne nécessite guère de capture d'écran car il se résume au simple fait de devoir faire attention à ce que vos données dans vos tables n'aient pas des espaces vides avant ou après les saisies sinon quoi vous aurez forcément des surprises!

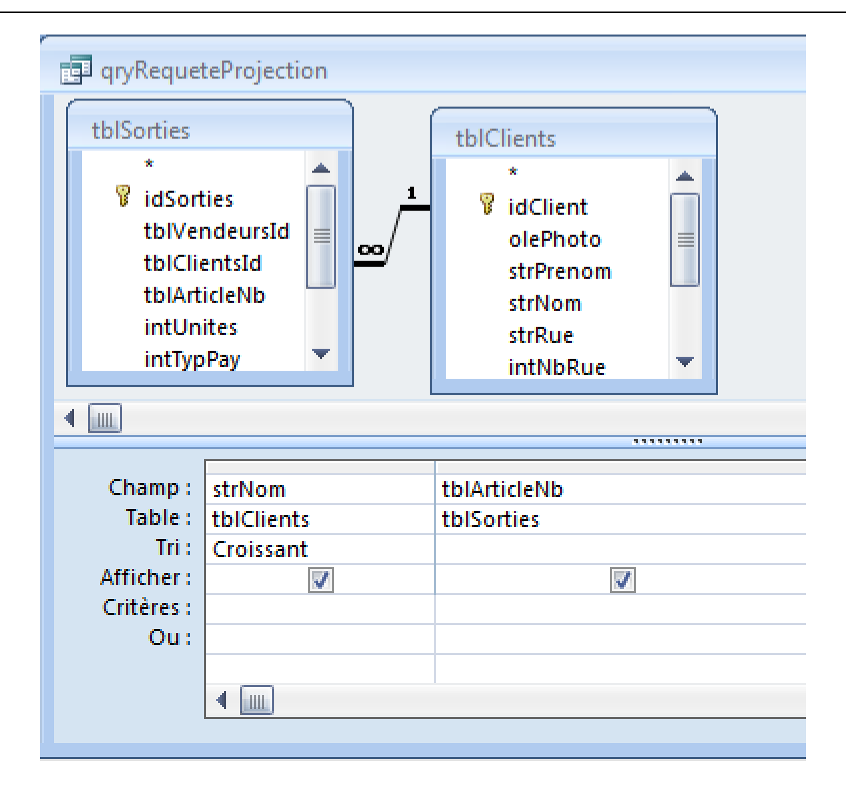
Le deuxième problème est un peut subtil. Considérons la requête de projection suivante:



et exécutons-la. Nous avons alors:



à la surprise de beaucoup d'utilisateurs MS Access, la colonne client n'est pas triée. Au fait **c'est normal** car le moteur de requête utilise le numéro de la clé primaire des clients provenant de la table *tblClients* pour faire le tri et non le nom des clients eux-mêmes! Il faut donc aller chercher la table *tblClients* pour la rajouter dans la requête et prendre le champ *strNom*:

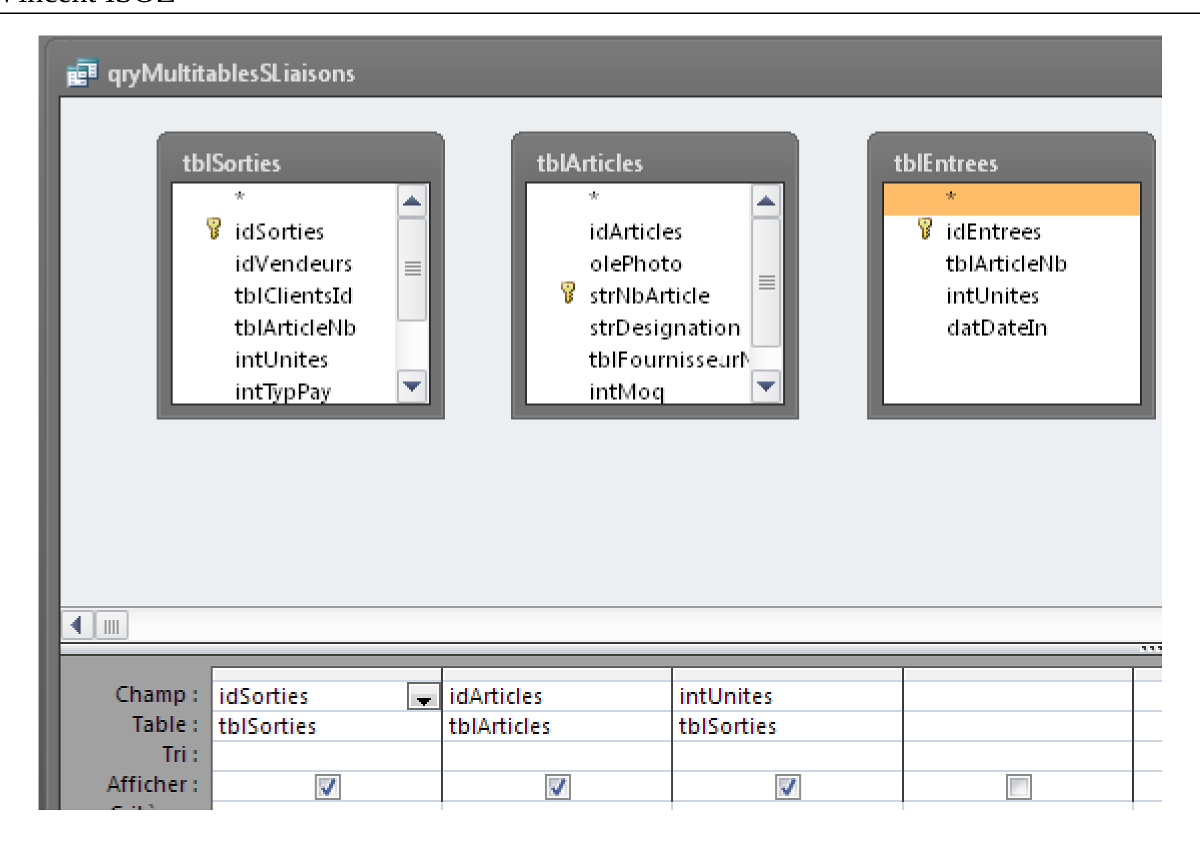


et dès lors le résultat sera conforme!

Observez ensuite ce qu'il se passe si vous changez le nom d'un champ (attribut) dans l'une des tables qu'utilise-la requête. Est-ce que la requête fonctionnera toujours?

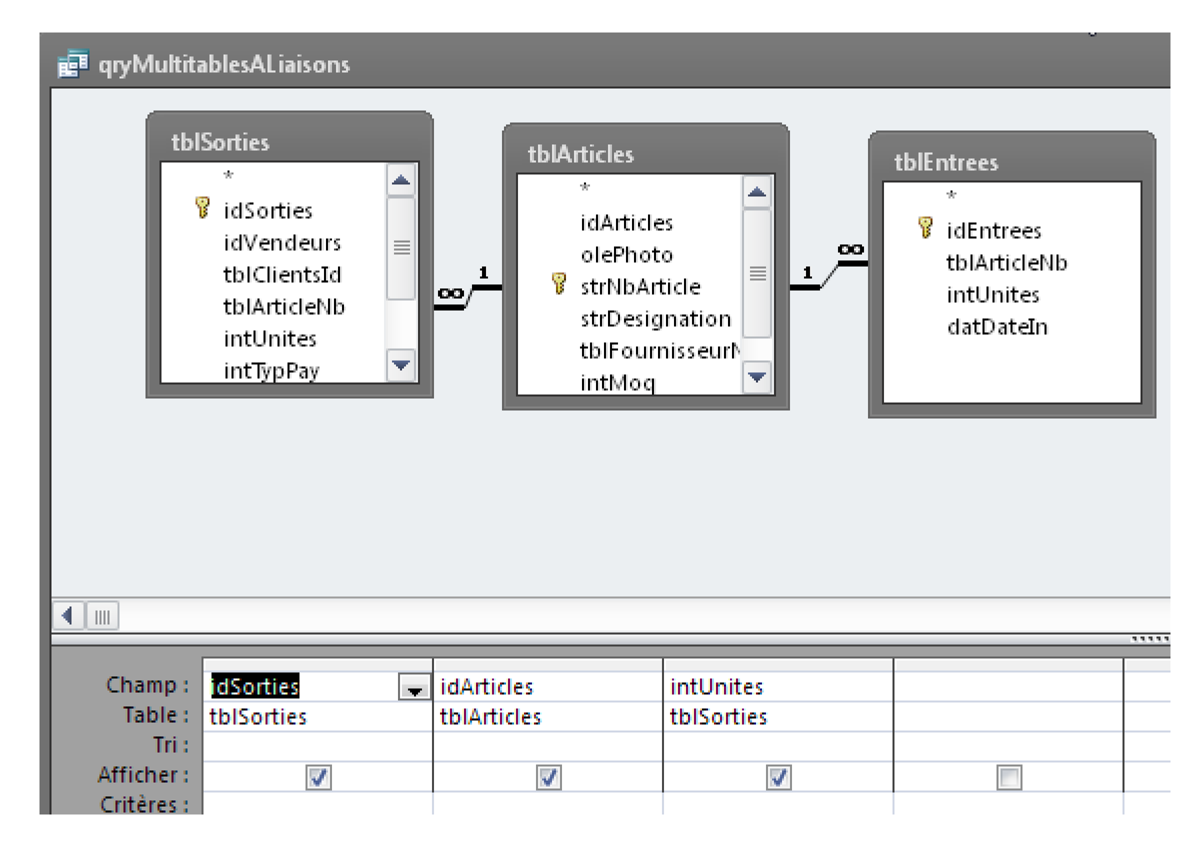
7.4 Requête multitable sans liaisons

Créez la requête suivante *qryMultitablesSLiaisons* et essayez d'argumenter le nombre de lignes obtenu dans le résultat par rapport au nombre de lignes chacune des tables respectives (est-il juste, est-il faux, pourquoi?, qu'est-ce qui se passe?):



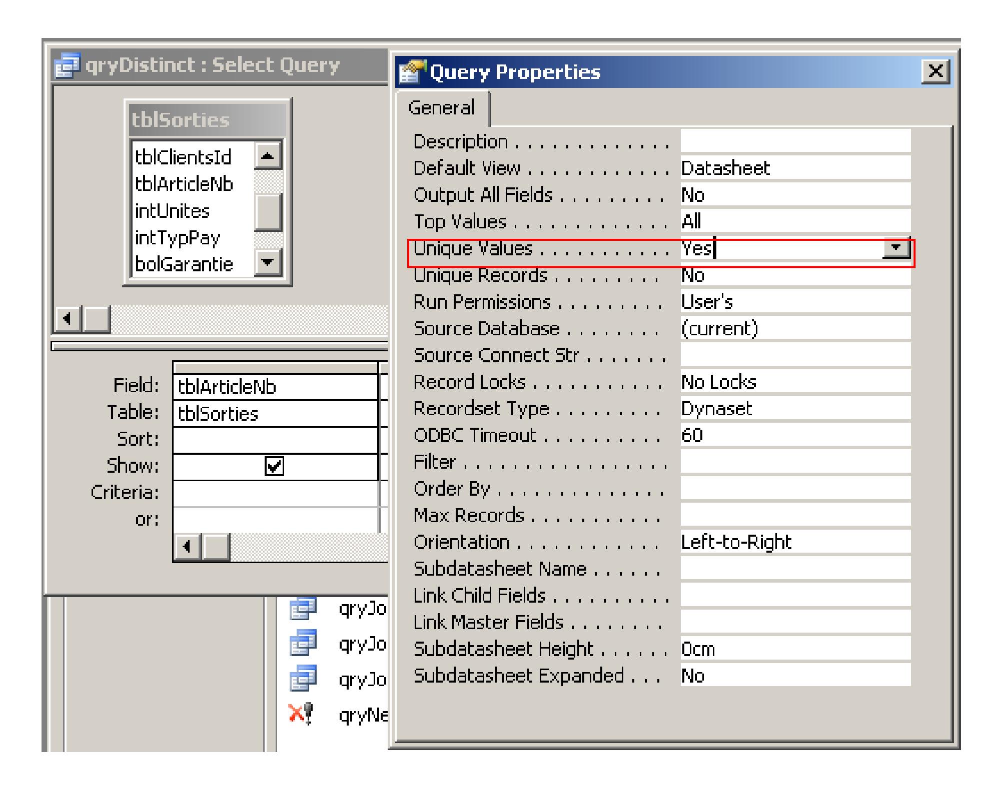
7.5 Requête multitable avec liaisons

Créez la requête suivante *qryMultitablesALiaisons* et essayez d'argumenter le nombre de lignes obtenues dans le résultat par rapport au nombre de lignes de chacune des tables respectives (est-il juste, est-il faux, pourquoi?, qu'est-ce qui se passe?):



7.6 Requête de distinction

Faites en sorte d'avoir une requête nommée *qryDistinction* qui affiche, sans doublons, la liste des articles vendus:



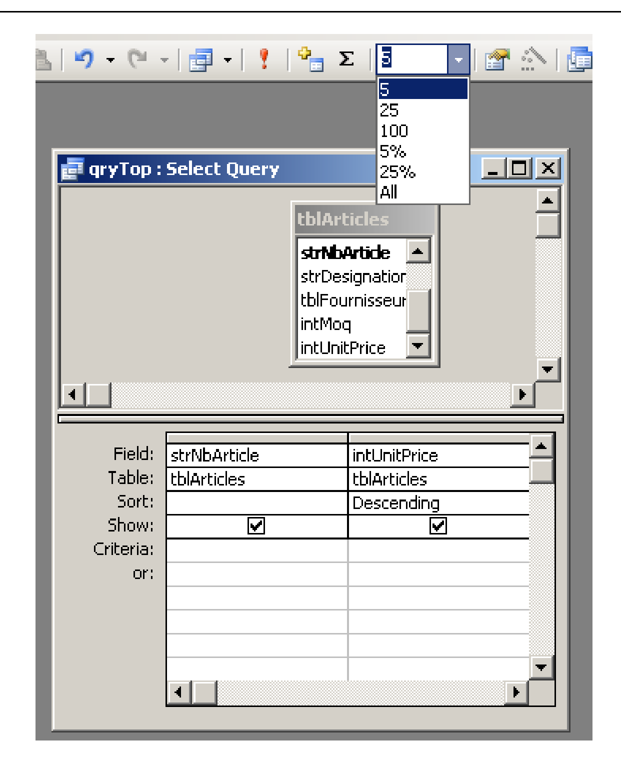
Remarque: Il existe une autre possibilité qui consiste à utiliser le regroupement comme va vous le montrer votre formateur (requête à enregistrer sous le nom de *qryDistinctionGroupee*), mais normalement cette dernière technique devrait plutôt être réservée pour des calculs arithmétiques.

Vérifiez si en créant un formulaire des deux requêtes précédentes, vous pouvez oui ou non créer de nouvelles données. Argumentez votre observation!

Remarque: *Unique Values* supprime tous les enregistrements à double des champs sélectionnés dans la requête (basée sur la commande SQL *DISTINCT*). *Unique records*, basée sur la commande SQL *DISTINCTROW*, supprime tous les enregistrements à double en prenant comme référence TOUTES les colonnes même si ces dernières ne sont pas toutes visibles dans la requête.

7.7 Cinq premiers

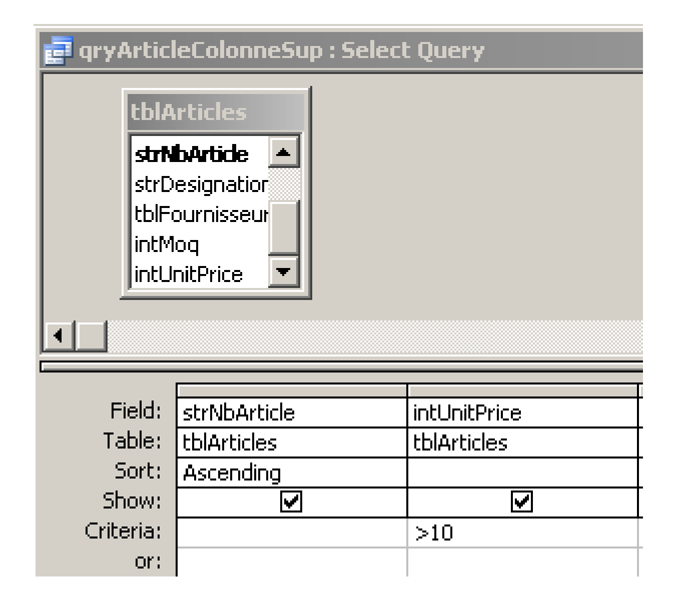
Créez une requête qui affiche le nom des articles et leur prix seulement (sous le nom *qryTop*) et faites en sorte que seulement les 5 plus coûteux articles apparaissent dans le résultat de la requête:



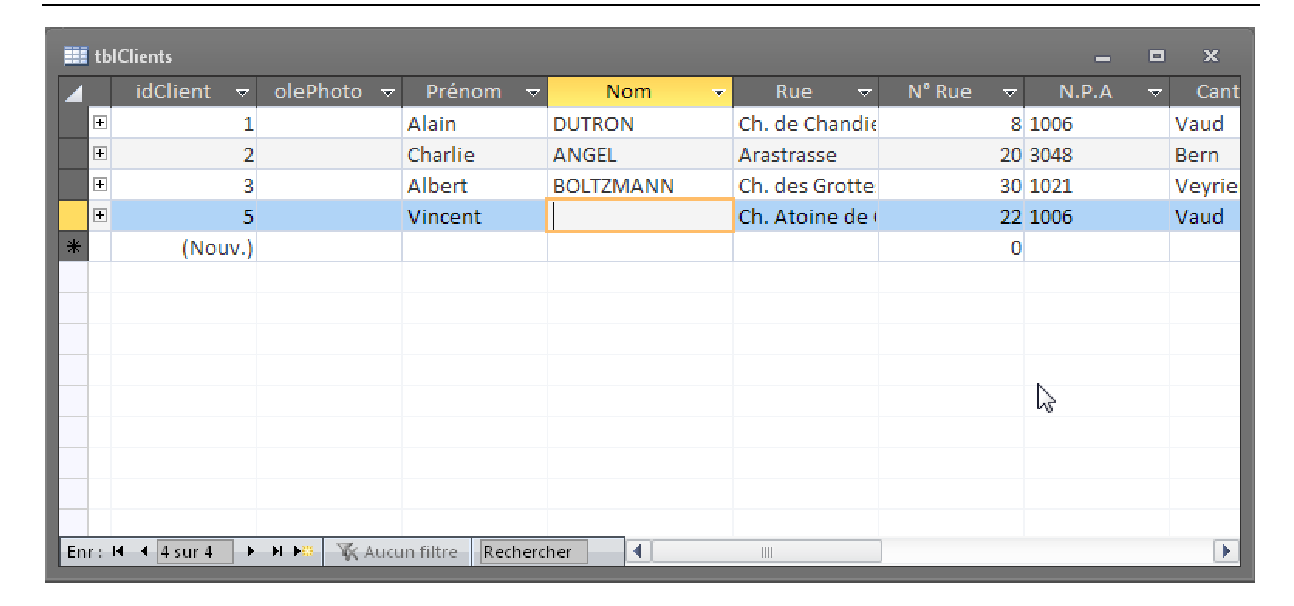
Remarque: le champ en question doit être au préalable trié si nous voulons les X premiers (ordre décroissants) ou X derniers (ordre croissant).

7.8 Requête avec critère

Modifiez manuellement la requête nommée *qryPrixArticles* afin qu'elle affiche uniquement les champs du numéro d'article et de prix unitaire de tous les articles dont le prix dépasse CHF 10.- (ne pas oublier de cliquer sur le bouton point d'exclamation pour lancer la requête). Créez un formulaire instantané, observez le résultat et fermez ce dernier sans l'enregistrer.



Pour le prochain exemple important considérons notre table des vendeurs:


tblClients

	idClient	olePhoto	Prénom	Nom	Rue	N° Rue	N.P.A	Cant
+	1		Alain	DUTRON	Ch. de Chandie	8	1006	Vaud
+	2		Charlie	ANGEL	Arastrasse	20	3048	Bern
+	3		Albert	BOLTZMANN	Ch. des Grotte	30	1021	Veyrie
+	5		Vincent		Ch. Atoine de (	22	1006	Vaud
*	(Nouv.)					0		

Enr : 4 sur 4 Aucun filtre Rechercher

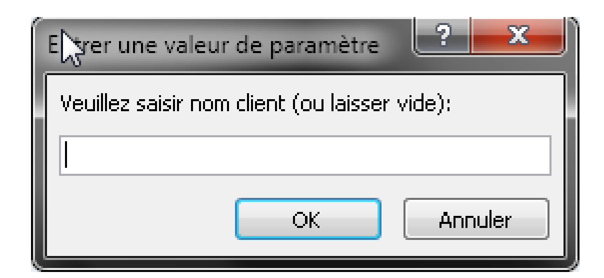
Si on crée la requête suivante:

qryLike

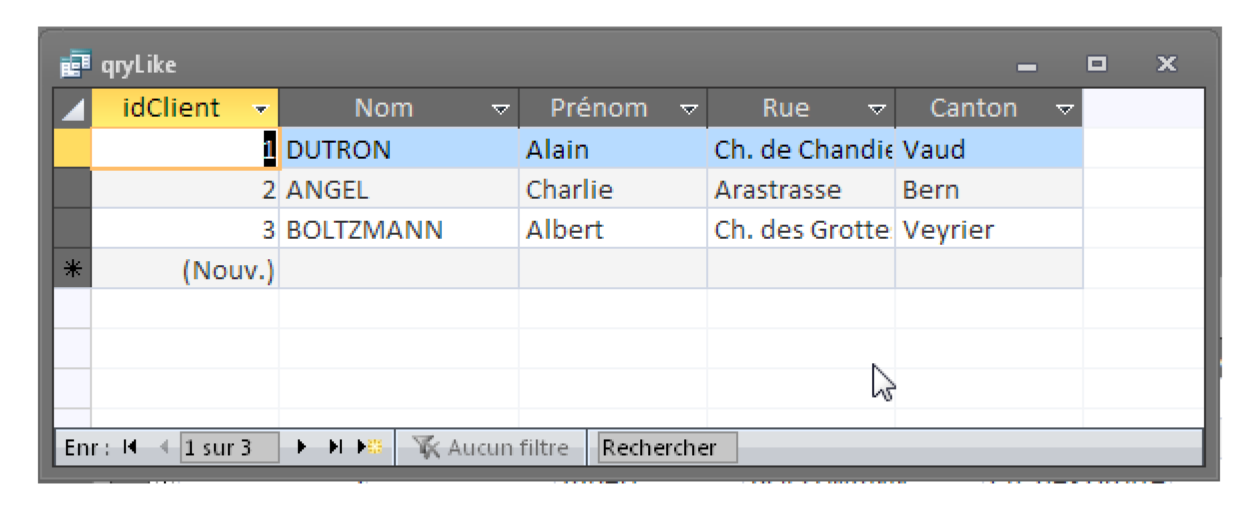
tblClients: olePhoto, strPrenom, strNom, strRue, intNbRue, strNPA, strCanton

Champ :	idClient	strNom	strPrenom	strRue	strCanton
Table :	tblClients	tblClients	tblClients	tblClients	tblClients
Tri :					
Afficher :	☑	☑	☑	☑	☑
Critères :		Comme "*" & [Veuillez saisir nom client (ou laisser vide):] & "*"			
Ou :					

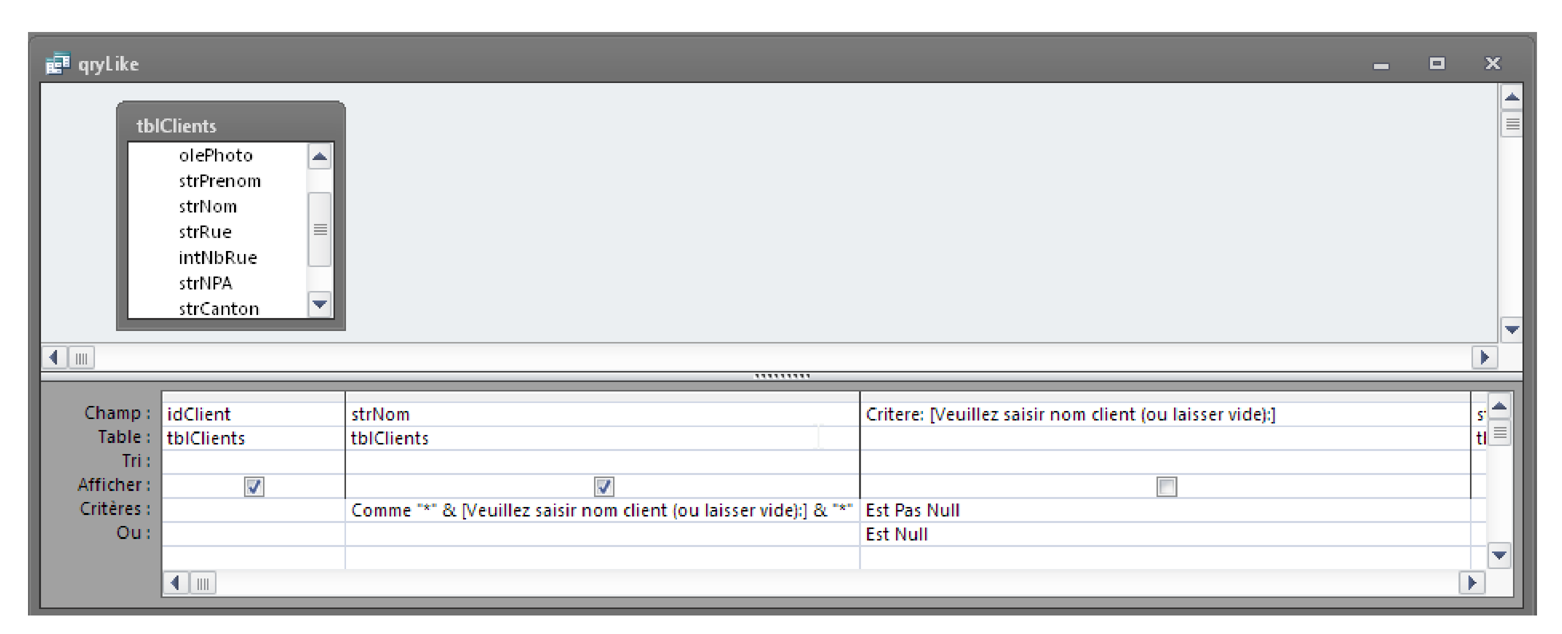
En exécutant:



et en laissant vide on obtient:

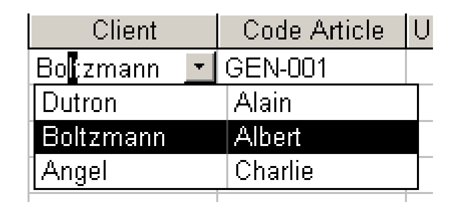


Donc tous les clients sauf ceux qui n'ont pas de *Nom*. Pour contourner cela, nous changeons notre requête comme suit:



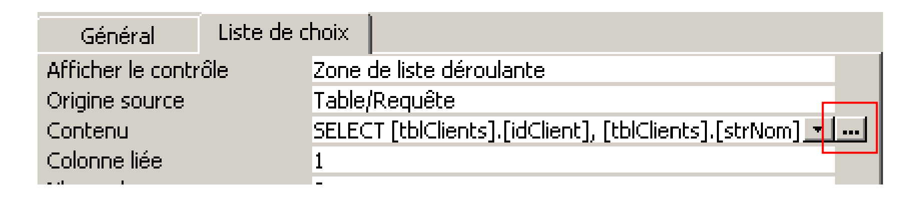
7.9 Requête concaténation dans liste de choix

Lorsque vous créez une liste de choix entre *tblSorties* et *tblClients* pour faire du même coup une relation, vous avez alors le résultat suivant:

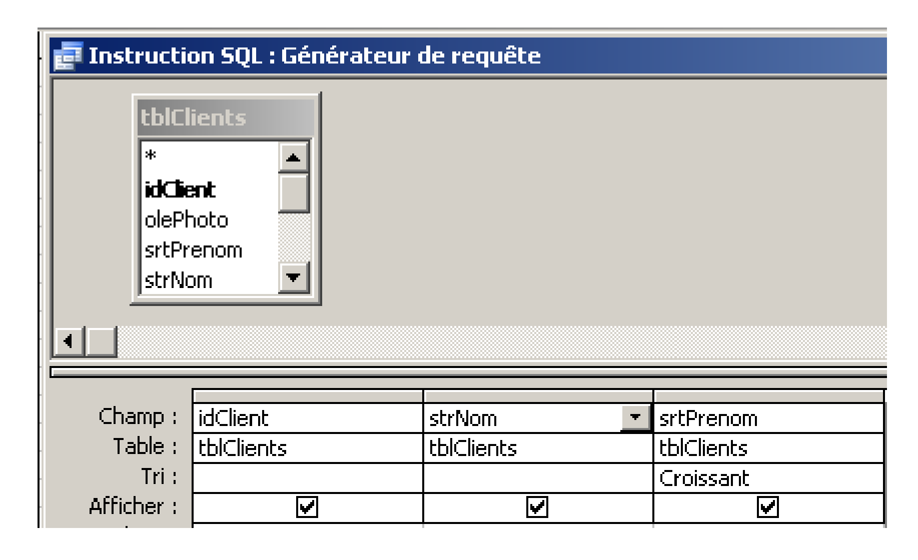


Le problème est alors lorsque la sélection d'un client est effectuée, seulement la première colonne de la liste de choix, apparaît dans la cellule de la table (respectivement dans le champ du formulaire.. pour les formulaires). Pour résoudre ceci, il suffit de faire une modification dans une requête qu'Access aura créée automatiquement pour vous lors de la création de la liste de choix

Effectivement, dans les paramètres de la liste, cliquez sur:



Apparaît alors:



Changez la requête comme indiqué ci-dessous:

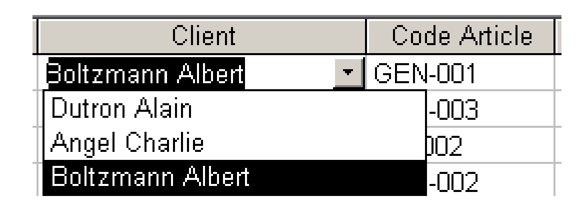
idClient	Client: [strNom] & " " & [strPrenom]
tblClients	
☑	☑

Ensuite, fermez la fenêtre de la requête (confirmez les changements) et de retour dans les propriétés de la liste choix, changez l'option *Nbre Colonnes* :

Contenu	SELECT [tblClients].[idClient], [tblClients].[strNom]
Colonne liée	1
Nbre colonnes	3

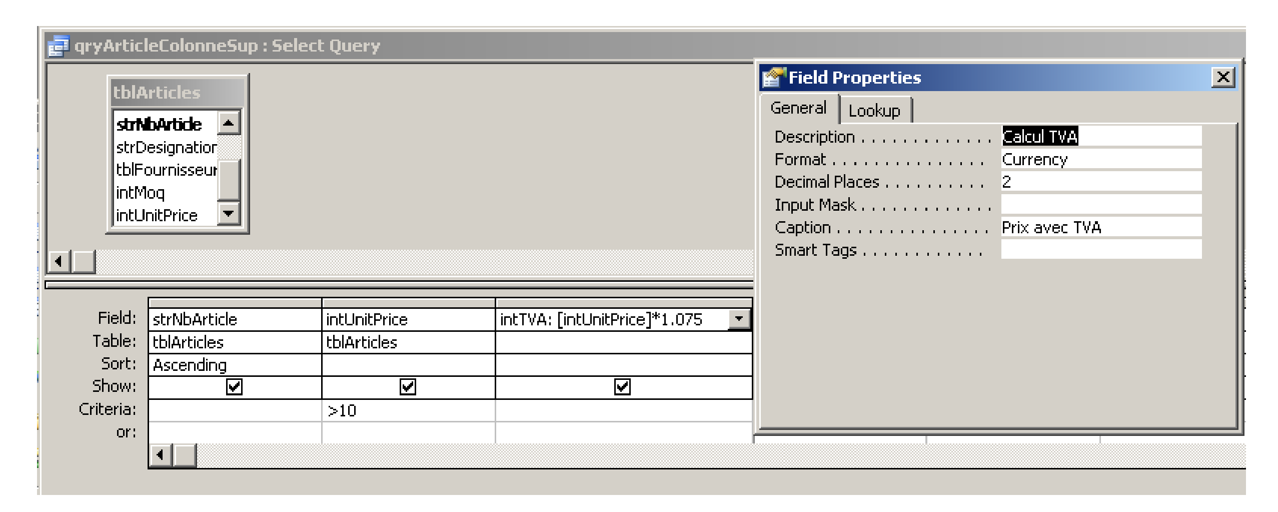
à la valeur 2 !

Vous obtenez alors le résultat suivant:



7.10 Colonne calculée

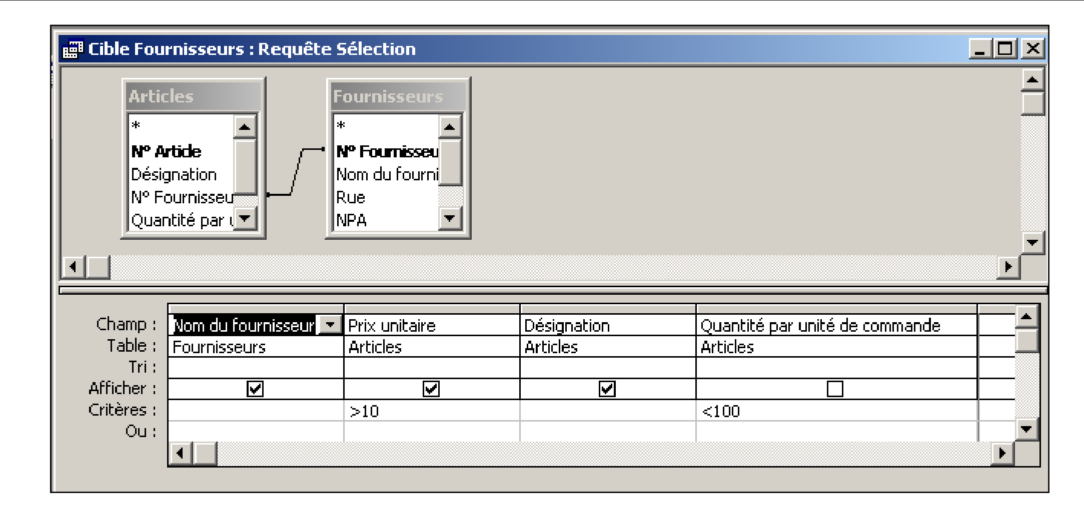
Créez une autre requête nommée *qryArticlesRequeteSpeciale* affichant tous les articles dont le prix dépasse 10.-. En plus des champs de numéro d'article et du prix unitaire, la table compilée devra contenir un nouveau champ calculé nommé *intTva* dans lequel la TVA (7.5%), appliquée sur les prix unitaires, sera calculée (mettre la colonne TVA au format monétaire en cliquant sur la formule avec le bouton droit de la souris: Propriétés/Format – au besoin exécuter la requête avant de définir l'option d'affichage de deux décimales).



7.11 Critères multiples

Créez une requête nommée *qryCibleFournisseurs* qui doit pouvoir permettre à un utilisateur de visualiser dans un formulaire les noms des fournisseurs qui ont un M.O.Q (Minimal Order Quantity) inférieur à 100 articles dont le prix par unité est supérieur à 10.-. Faites également en sorte que ces fournisseurs soient triés dans l'ordre croissant du nombre de quantités commandées. En plus, on doit voir dans le formulaire qui ressortira de cette requête, uniquement le nom du fournisseur, le prix unitaire de ses articles et leur désignation.

Pour faire cet exercice il vous faudra insérer les deux tables *tblFournisseurs* et *tblArticles* dans la requête (pour rajouter un table cliquer sur le bouton)



Champ :	Nom du fournisseur	Prix unitaire	Désignation	Quantité par unité de commande
Table :	Fournisseurs	Articles	Articles	Articles
Tri :				
Afficher :	☑	☑	☑	☐
Critères :		>10		<100
Ou :				

Comme vous pouvez le voir sur la figure ci-dessus, les deux tables sont reliées entre elles. Avant de voir en quoi cela consiste, exécutez votre requête sans la liaison, créez-en un formulaire instantané que vous enregistrerez sous le nom *frmCibleFournisseurs* et essayez d'en tirer une conclusion. Vous verrez que la liaison permet de faire barrière à cette absurdité (nous verrons plus loin, plus en détail, les différentes possibilités).

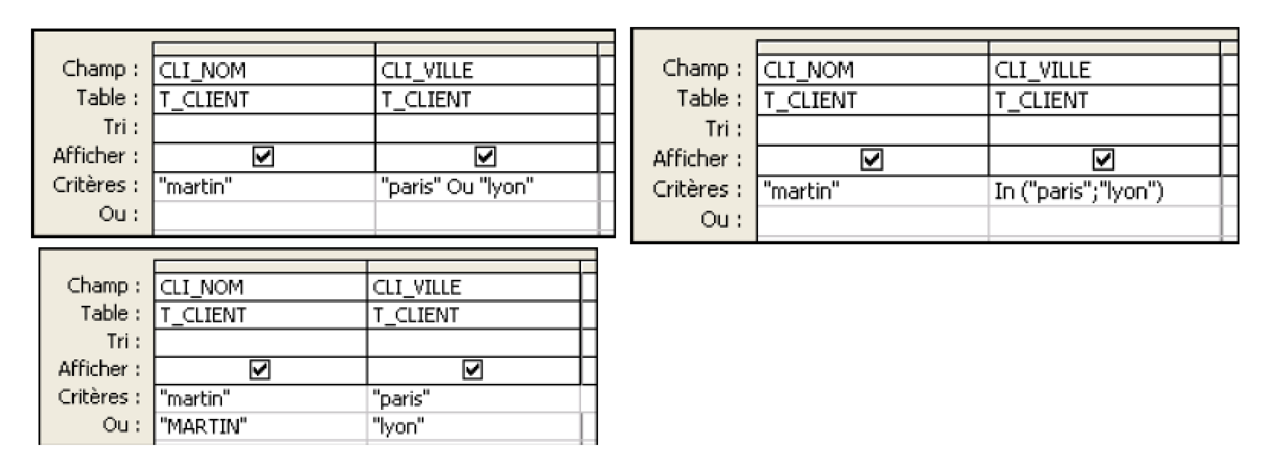
Si vous essayez de saisir des données par l'intermédiaire du formulaire Cible Fournisseurs vous verrez que l'opération d'ajout d'enregistrements vous sera refusée. Expliquez pourquoi.

Remarque: Attention à l'utilisation de la ligne OU, elle implique un doublement des critères pour fonctionner:

Field:	datDateIn	strNbArticle	strNom	intUnitPrice
Table:	tblEntrees	tblArticles	tblFournisseurs	tblArticles
Sort:				
Show:	☑	☑	☑	☑
Criteria:		"INF-001"		>1 And <5
or:		"INF-001"		>20

Remarque: Il s'agit du premier exemple utilisant deux tables et il faut savoir que dans le cadre des requêtes que l'on ne doit jamais avoir une table non liée (au cas où il y en aurait plusieurs bien sûr...).

Il existe souvent plusieurs manières d'écrire un même critère. Par exemples les trois requêtes suivantes sont équivalentes:

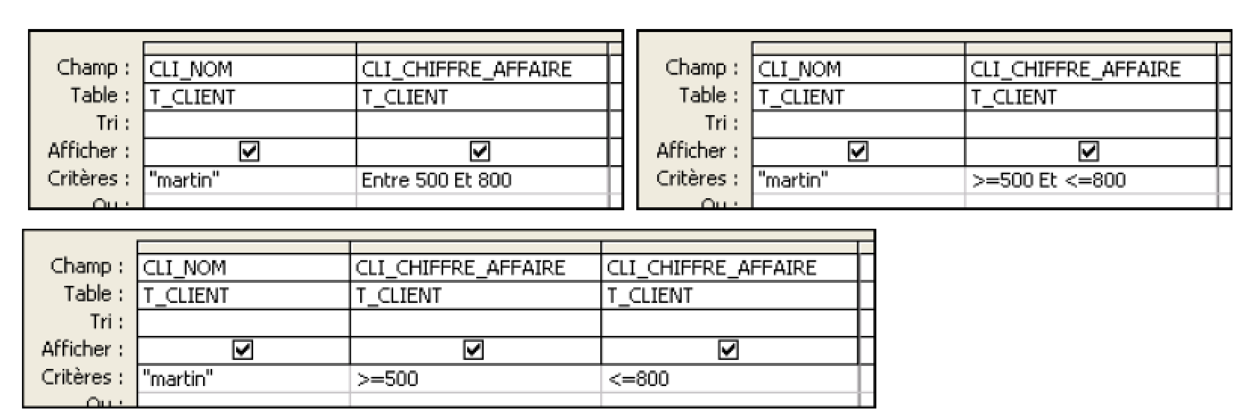


Champ :	CLI_NOM	CLI_VILLE
Table :	T_CLIENT	T_CLIENT
Tri :		
Afficher :	☑	☑
Critères :	"martin"	"paris" Ou "lyon"
Ou :		

Champ :	CLI_NOM	CLI_VILLE
Table :	T_CLIENT	T_CLIENT
Tri :		
Afficher :	☑	☑
Critères :	"martin"	In ("paris";"lyon")
Ou :		

Champ :	CLI_NOM	CLI_VILLE
Table :	T_CLIENT	T_CLIENT
Tri :		
Afficher :	☑	☑
Critères :	"martin"	"paris"
Ou :	"MARTIN"	"lyon"

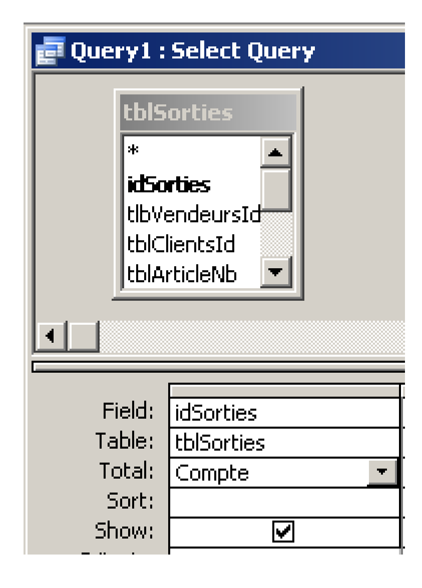
Il en va de même pour les trois suivantes:



7.12 Calcul de synthèse (d'agrégation)

Nous souhaitons compter le nombre de ventes effectuées par l'ensemble des vendeurs à l'aide de la table *tblSorties*.

Créez pour cela la requête suivante en cliquant lors du mode création sur le bouton Σ pour faire apparaître le champ *Total*:

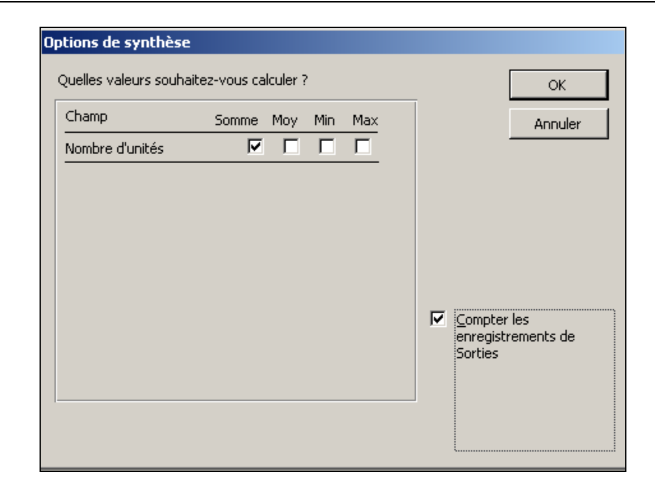


7.13 Regroupement et calculs

Le but maintenant est de créer un formulaire (basé sur une requête nommée *qrySommeSorties*), qui va permettre à l'utilisateur de connaître combien d'articles d'un certain type ont été vendus et ce en combien de fois.

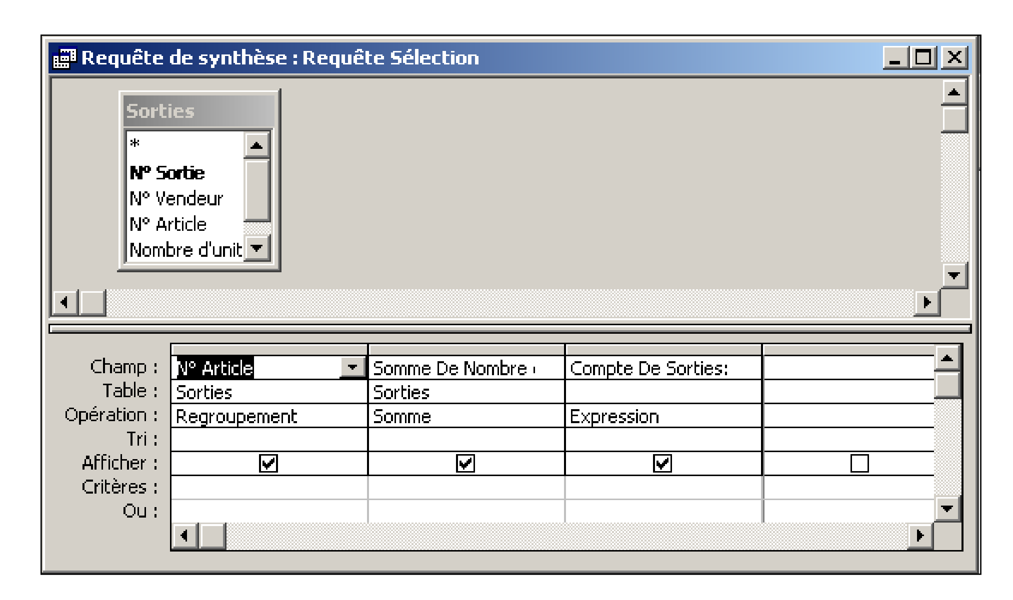
Pour cela, il vous faut créer une requête basée sur la table *tblSortie* dont on prendra le champ *idSortie, idArticle, intNbUnites*

Une fois que vous aurez choisi les champs, il faudra choisir dans l'assistant le bouton radio intitulé "De synthèse" et cliquer sur le bouton "Options de synthèse". Validez ensuite les champs *Somme* et *Compter les enregistrements de Sorties*.

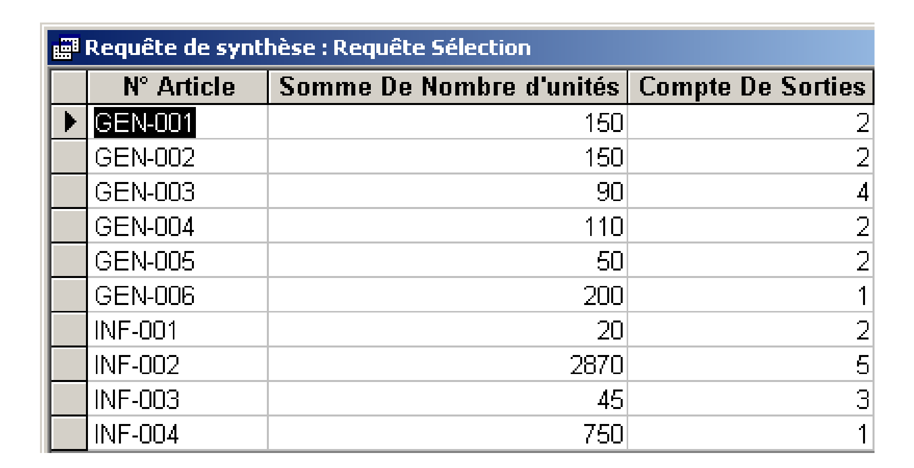


Si vous terminez l'assistant et exécutez la requête, vous y trouverez une "erreur" (introduite exprès par le formateur dans l'énoncé de l'exercice). Quelle en est l'origine? Comment la corriger?

Le résultat définitif est le suivant:



Si vous exécutez la requête, vous devriez obtenir le résultat suivant:


Requête de synthèse : Requête Sélection

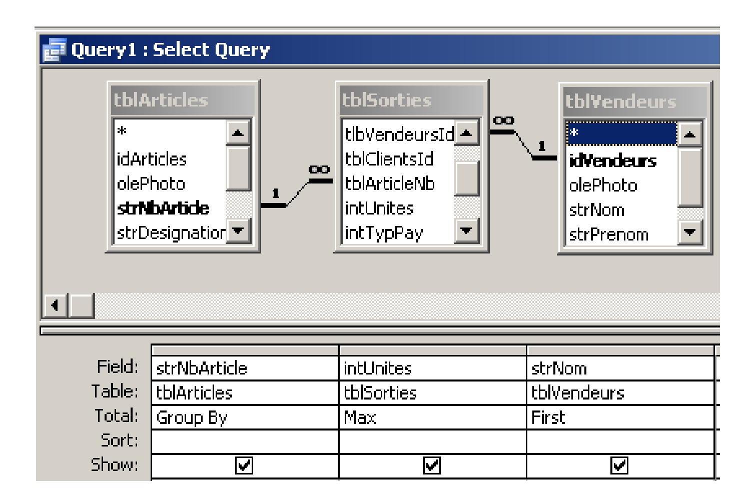
N° Article	Somme De Nombre d'unités	Compte De Sorties
GEN-001	150	2
GEN-002	150	2
GEN-003	90	4
GEN-004	110	2
GEN-005	50	2
GEN-006	200	1
INF-001	20	2
INF-002	2870	5
INF-003	45	3
INF-004	750	1

Si vous comparez avec le contenu de la table *tblSorties*, vous verrez que la synthèse est bien correcte.

Exportez le contenu de cette requête dans MS Excel et observez ce qui se passe pour le champ du code d'article (vous verrez que celui-ci apparaît normalement puisque le code fait lui-même office de clé primaire).

En tant qu'exercice créez une requête que vous nommerez *qryPerformance* qui affiche pour un article donné, le vendeur qui en a vendu le plus et la quantité correspondante (si deux vendeurs sont à égalité, nous prendrons le premier dans l'ordre de saisie).

Solution:



Soit:

Code Article	MaxOfintUnites	FirstOfstrNom
GEN-001	100	Weidman
GEN-002	1000	Butty
GEN-003	200	Castrini
GEN-004	80	Barbier
GEN-006	200	Weidman
INF-001	15	Massa
INF-002	1500	Clerc
INF-003	30	Hoffmann
INF-004	750	Butty

Attention!! N'utilisez jamais les regroupements dans le but d'avoir une table à valeurs uniques dont l'objectif serait d'être utilisés dans des listes déroulantes de formulaires. Effectivement, ce type de stratégie vous posera des problèmes pour écrire avec des macros du type *DéfinirValeur*. Pour faire en réalité proprement une liste d'éléments uniques, il vous faut utiliser l'option *Unique* que nous verrons comme exemple plus loin.

7.14 Requête et macro d'export

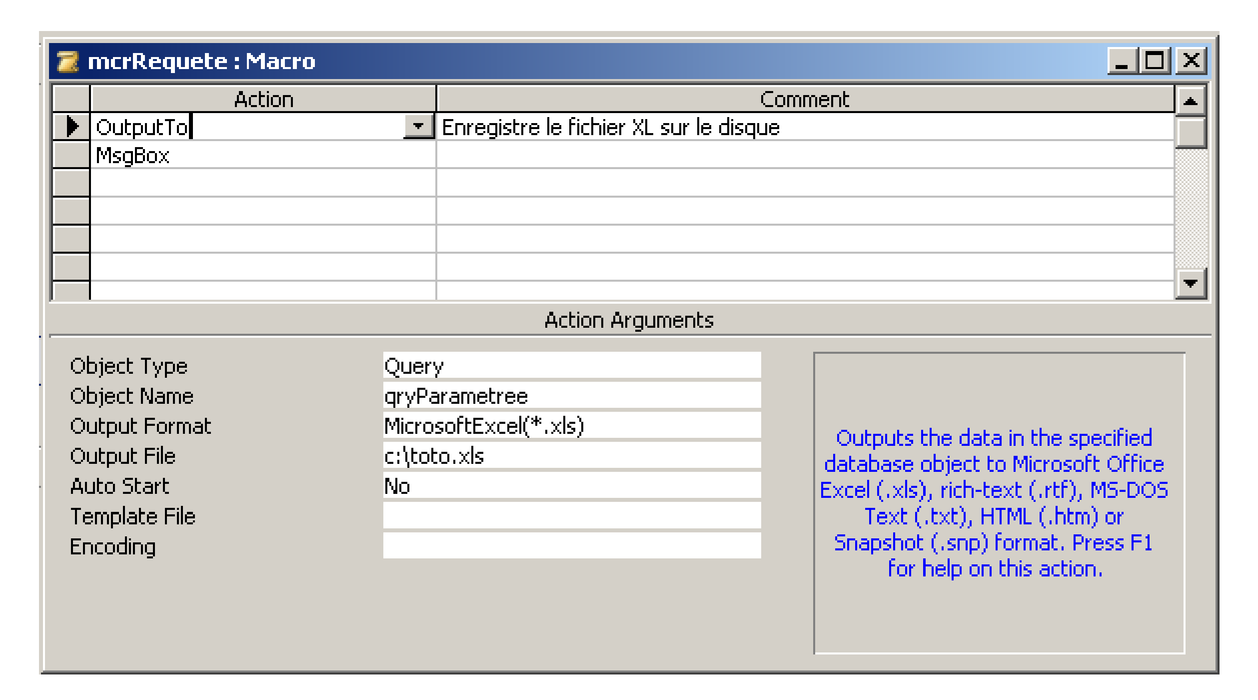
Créez une requête *qryPiege* basée sur la table *tblSorties* en prenant tous les champs et toutes les données *.

Créez **ensuite** une macro pour l'export vers MS Excel de cette requête (voir figure ci-dessous)

Exportez d'abord la table *tblSorties vers* vers MS Excel manuellement et faites ensuite l'export de la requête *qryPiege* avec la macro.

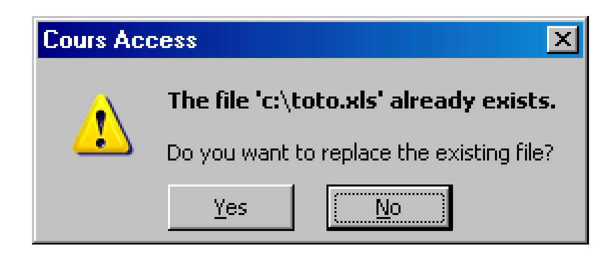
Quelle différences pouvez-vous observer entre l'export manuel et l'export avec la macro au niveau du tableau MS Excel obtenu?

Remarque: Si vous avez Access 2007 ou ultérieur, exportez au format *.xlsx sinon quoi il peut y avoir une mauvais export de données (incomplet).

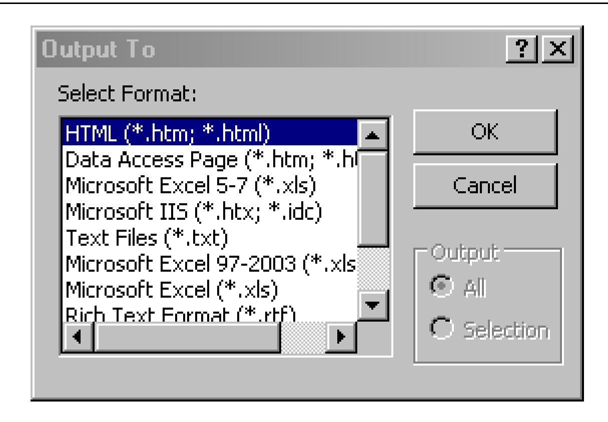


Remarque: En français l'équivalent de *OutputTo* est *CopierVers*

Si le fichier généré existe déjà à l'emplacement spécifié, MS Access vous demandera si vous souhaitez l'écraser:

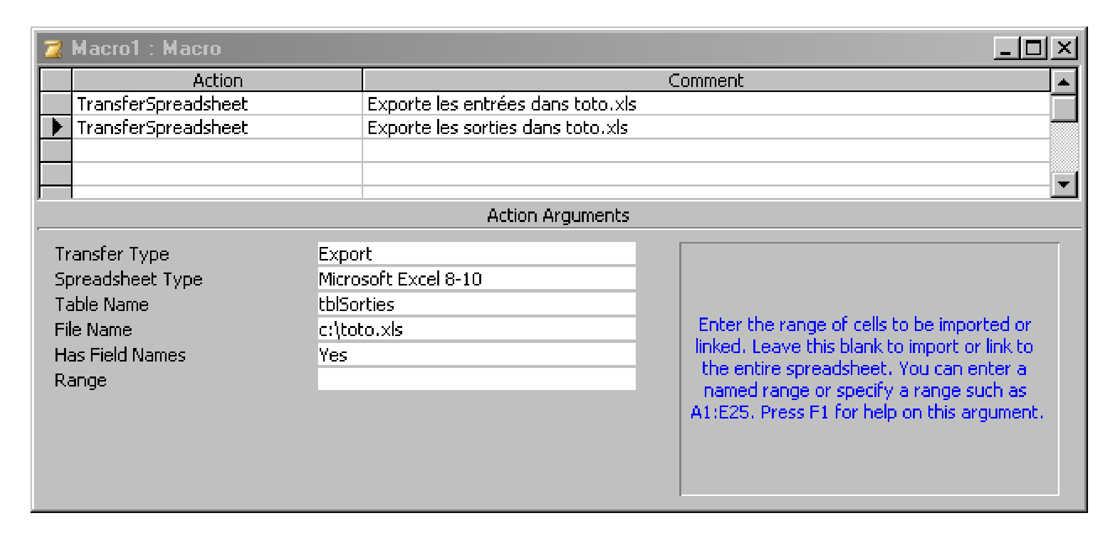


Si vous ne spécifiez pas de format, lors de l'exécution de la macro, le système demandera à l'utilisateur sous quel format il souhaite l'exporter:

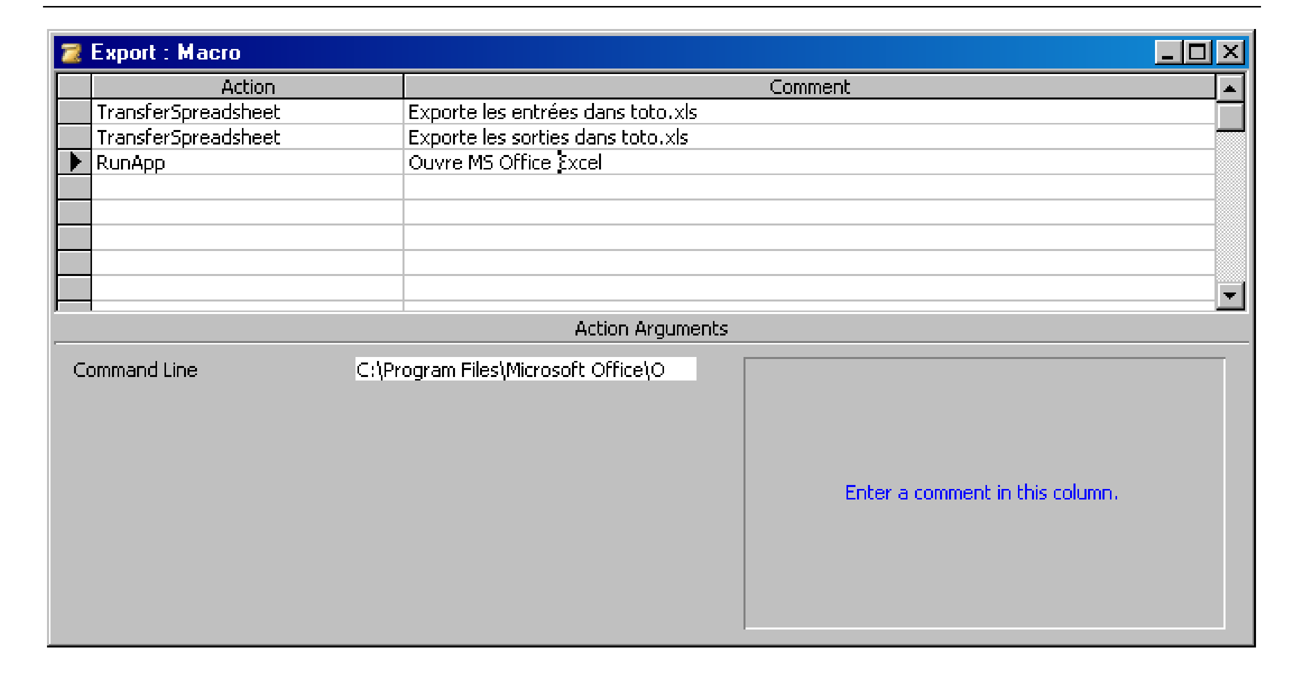


Que constatez-vous pour les champs clients et vendeurs ? Que peut-on en conclure ? Quelles sont les conséquences après coup ?

Si vous souhaitez pouvoir exporter plusieurs requêtes ou tables dans un même fichier MS Excel cela ne fonctionnera pas avec *Output To*, il vous faudra utiliser l'action suivante *TransferSpreadsheet* (TransférerFeuilleCalcul):



et comme *TransfertSpreadsheet* n'ouvre pas MS Excel automatiquement il suffit de rajouter l'action *RunApp*:



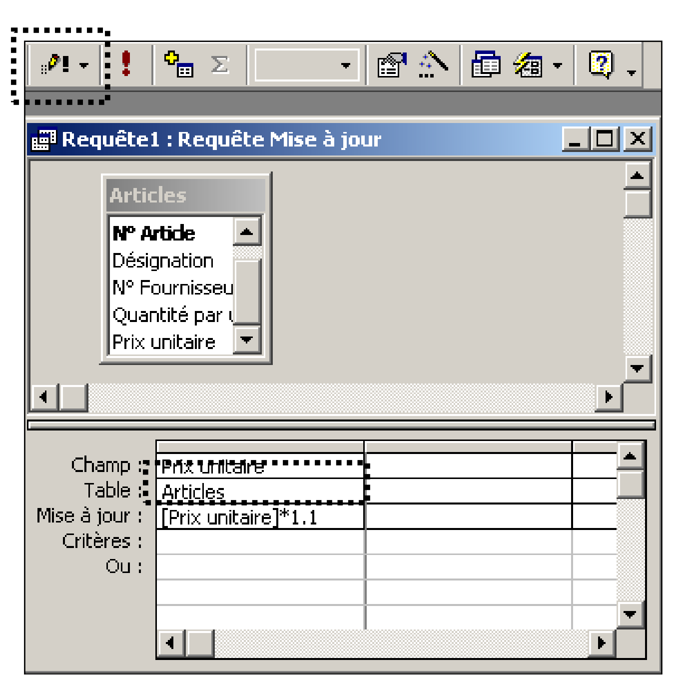
ou dans la *Command Line* (*ExécuterApplication*) nous avons typiquement:

C:\Program Files\Microsoft Office\OFFICE11\excel.exe c:\toto.xls

Remarque: Depuis MS Access 2007 et 2010, si vous exportez une table ou requête contenant un champ date dont la colonne n'est pas assez grande pour afficher toute l'information (vous aurez donc des: ######), vous vous retrouverez avec un fichier MS Excel où il sera malheureusement écrit physiquement ####... ce qui constitue bien évidemment un bug. Il faudra alors au préalable adapter la colonne de votre table ou requête de manière adéquate.

7.15 Requête mise-à-jour

Nous allons maintenant créer une requête de "Mise à jour" qui va augmenter le prix de tous les articles de la table *tblArticles* de 10%.



Ce qui est dommage, c'est que si l'on laisse la formule comme représentée ci-dessus, l'utilisateur lambda ne pourra pas choisir la valeur de l'augmentation. Pour ajouter un peu d'interactivité, nous allons plutôt écrire la chose suivante dans le champ "Mise à jour":

[Prix unitaire](1+[Saisissez l'augmentation en %]/100)*

Enregistrez cette requête sous le nom "Requête mise à jour".

Remarques:

R1. A chaque fois que vous exécutez une requête d'action un message d'avertissement apparaîtra à l'écran. Pour désactiver celui-ci sans faire de VBA allez dans le menu *Outils/Options* et dans l'onglet *Modifier/Rechercher* désactivez la case à cocher *Requêtes d'action* . Sinon avec du VBA il suffit d'utiliser la commande docmd.setwarnings false. Il se peut cependant que ni la première, ni la deuxième solution ne fonctionnent, dès lors il vous faudra écire le code VBA suivant:

```
Dim strSQL as String
strSQL = "Name of saved query or sql in vba"
CurrentDb.Execute strSQL, dbFailOnError
```

où c'est le dbFailOnError qui joue ici un rôle majeur pour se débarasser de tout message d'erreur!

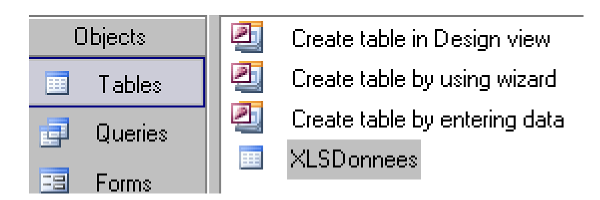
R2. Si vous sélectionnez l'astérisque (*) dans une requête, cela présente un avantage par rapport à l'opération qui consiste à sélectionner tous les champs. Lorsque vous utilisez l'astérisque, les résultats de la requête comprennent automatiquement tous les champs qui ont été rajoutés à la table (pendant son évolution) ou à la requête sous-jacente (si la requête est créée à partir d'une requête) après la création de la requête et excluent automatiquement les champs qui sont supprimés. Lorsque vous utilisez l'astérisque, vous ne pouvez pas trier les enregistrements ou spécifier les critères de champs sauf si vous ajoutez ces champs ensuite à part dans la grille de création de la requête et que vous désactivez son affichage lors de l'exécution de la requête.

R3. Si vous avez des calculs avec des champs "null" à effectuer (ce qui n'est pas équivalent au nombre 'nul' = 0) les résultats ne sortent pas. Il faut dès lors dans le champ contenant la relation mathématique ajouté à l'endroit adéquat une des deux formules suivantes:

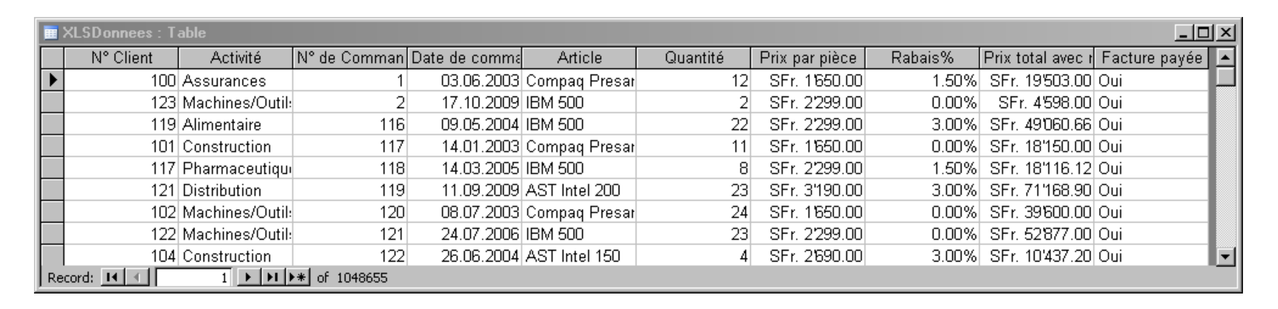
- IIf(isnull([nomchamp]);0;[nomchamp])
- nz([nomchamp];0) // convertit la valeur du champ 'nomchamp' en la valeur du second argument de la fonction

7.16 Requête mise-à-jour (Rechercher/Remplacer)

Pour cette requête (dont nous avons déjà fait mention au début de cet e-book), le lecteur devra ouvrir la base de données Access Données 95-2003.mdb que mettra le formateur à disposition et ouvrir la seule table disponible:



qui contient donc 1'048'655 lignes:

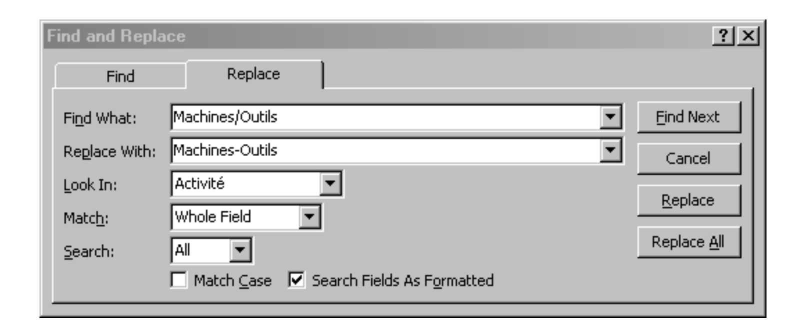


XLSDonnees : Table

N° Client	Activité	N° de Comman	Date de comma	Article	Quantité	Prix par pièce	Rabais%	Prix total avec r	Facture payée
100	Assurances	1	03.06.2003	Compaq Presar	12	SFr. 1'650.00	1.50%	SFr. 19'503.00	Oui
123	Machines/Outil:	2	17.10.2009	IBM 500	2	SFr. 2'299.00	0.00%	SFr. 4'598.00	Oui
119	Alimentaire	116	09.05.2004	IBM 500	22	SFr. 2'299.00	3.00%	SFr. 49'060.66	Oui
101	Construction	117	14.01.2003	Compaq Presar	11	SFr. 1'650.00	0.00%	SFr. 18'150.00	Oui
117	Pharmaceutiqu	118	14.03.2005	IBM 500	8	SFr. 2'299.00	1.50%	SFr. 18'116.12	Oui
121	Distribution	119	11.09.2009	AST Intel 200	23	SFr. 3'190.00	3.00%	SFr. 71'168.90	Oui
102	Machines/Outil:	120	08.07.2003	Compaq Presar	24	SFr. 1'650.00	0.00%	SFr. 39'600.00	Oui
122	Machines/Outil:	121	24.07.2006	IBM 500	23	SFr. 2'299.00	0.00%	SFr. 52'877.00	Oui
104	Construction	122	26.06.2004	AST Intel 150	4	SFr. 2'690.00	3.00%	SFr. 10'437.20	Oui

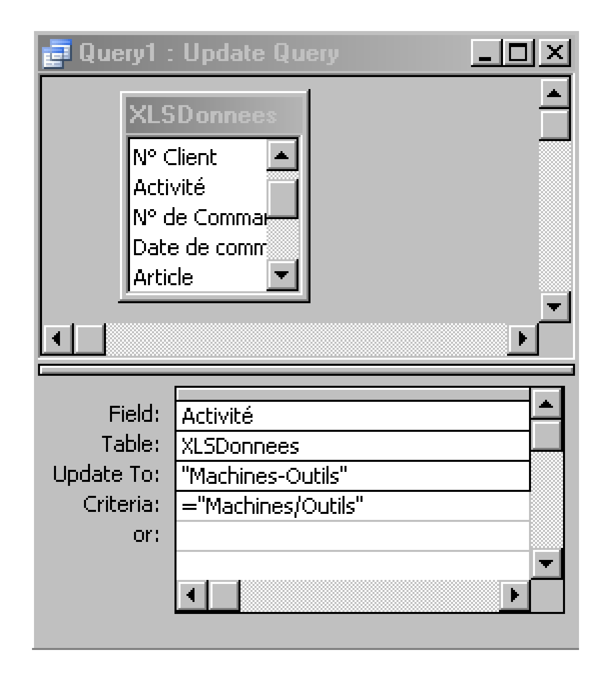
Record: 1 of 1048655

Si nous souhaitons faire un Rechercher/Remplacer de "Machines/Outils" en "Machines-Outils":

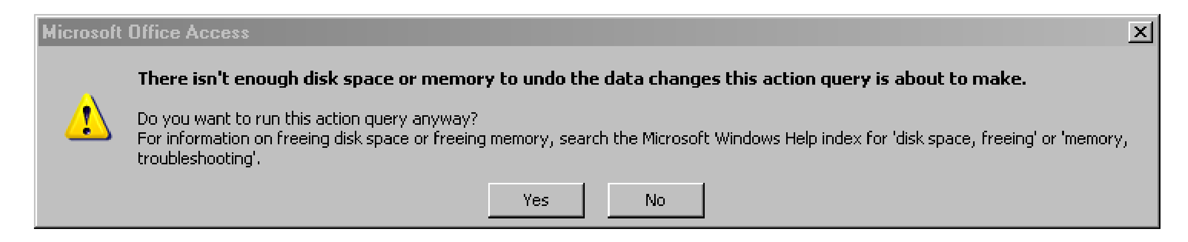


Il fera le remplacement que pour les 10'000 premier environ…

Pour pallier à ceci il suffit de créer la requête de mise-à-jour suivante:



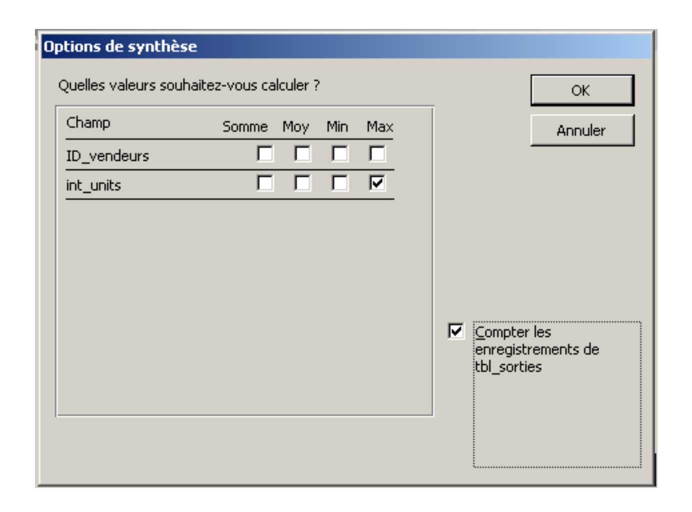
et d'exécuter… vous aurez alors le message d'avertissement (logique…) suivant:



qu'il suffira de valider par *Yes*.

7.17 Requêtes de synthèse

Créez à l'aide de l'assistant requête simple de synthèse et de vos connaissances sur les formulaires, une requête de synthèse (et non pas une requête d'analyse croisée !!!) qui donnera (affichera) à l'utilisateur depuis la table *tblSorties* les vendeurs qui dans l'ordre décroissant ont fait la plus grande vente (non pas le cumul ! mais la plus grande vente unique) et ce en indiquant sur combien de ventes tel que les options choisies dans l'assistant soient les suivantes:



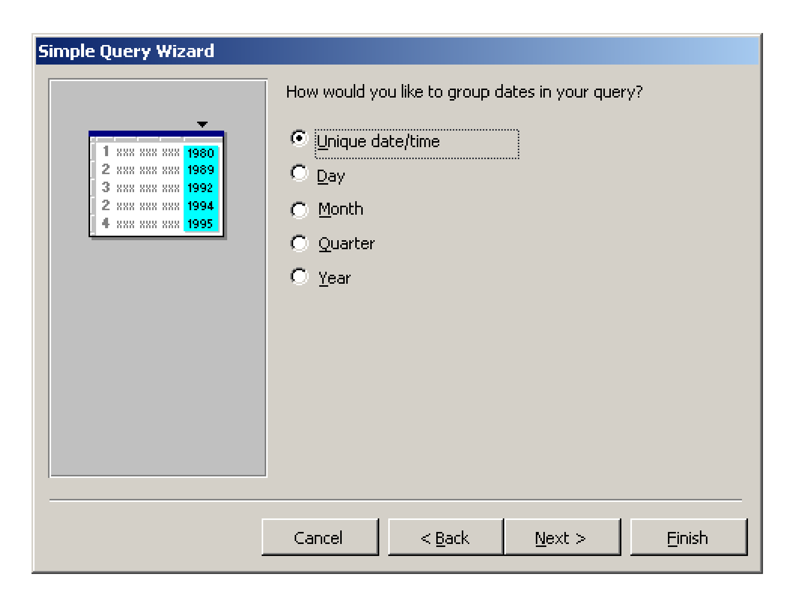
Le tableau généré par la requête devra ressembler à quelque chose comme cela:

Vendeurs	Max De int_units	Compte De tbl_sorties
Clerc	1500	4
Mettraux	750	2
Butty	750	3
Massa	500	3
Weidman	200	2
Hoffmann	100	2
Barbier	80	1
Castrini	30	3
Mettrez	20	3
De Siebental	5	1

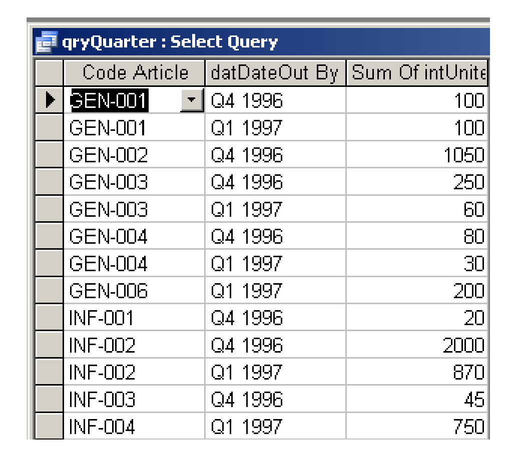
L'objectif est de créer un formulaire qui n'affichera que le premier vendeur de la liste (sans que l'utilisateur puisse voir les deuxième, troisième,… vendeurs).

Il est bien sûr facile d'indiquer à l'utilisateur seulement le premier vendeur. Pour cela, sans passer par le VBA, il suffit dans les propriétés du formulaire et de désactiver le navigateur d'enregistrements.

Avec l'assistant de requêtes, créez maintenant une requête de synthèse de la table *tblSorties* (comprenant les champs *strNbArticles*, *intUnites*, *datDateOut*) qui vous indique la somme des articles vendus par trimestre:



Vous devriez obtenir le résultat suivant:


qryQuarter : Select Query

Code Article	datDateOut By	Sum Of intUnite
GEN-001	Q4 1996	100
GEN-001	Q1 1997	100
GEN-002	Q4 1996	1050
GEN-003	Q4 1996	250
GEN-003	Q1 1997	60
GEN-004	Q4 1996	80
GEN-004	Q1 1997	30
GEN-006	Q1 1997	200
INF-001	Q4 1996	20
INF-002	Q4 1996	2000
INF-002	Q1 1997	870
INF-003	Q4 1996	45
INF-004	Q1 1997	750

7.18 Requête d'union

1. Créez une requête de création de table qui sauvegarde toutes les sorties comprises entre deux dates dans une table nommée *tblBackUp* et exportez ensuite cette sauvegarde dans MS Excel. Si possible, automatisez le tout avec une requête paramétrée (afin que nous puissions choisir l'année) et une macro.

2. Importez ou créez dans votre base les tables de sauvegardes des trois premières années du magasin (1978, 1979,1980) qui sont dans le fichier *BackUpSorites.xls* et nommez les respectivement: *tblSorties78*, *tblSorties79*, *tblSorties80*

Remarque: Comme vous le verrez, les ventes annuelles étaient maigres au début….

La table de 1978 (en faisant un import et sans oublier de supprimer après l'import à la main les lignes vides):

tblSorties78 : Table

idSorties	VendeursId	ClientsId	strNbArticle	intUnites	bolTypPay	datDateOut
1	1	3	GEN-001	50	-1	28.11.1978
2	3	2	INF-002	10	0	10.12.1978
3	8	1	GEN-004	1500	0	10.12.1978
4	7	1	INFO-001	100	-1	11.12.1978

La table 1979 (en faisant un import avec liaison de la table MS Excel):

tblSorties79 : Table

idSorties	VendeursId	ClientsId	strNbArticle	intUnites	bolTypPay	datDateOut
1	2	3	GEN-002	24	-1	28.11.1979
2	4	2	GEN-001	43	0	10.12.1979

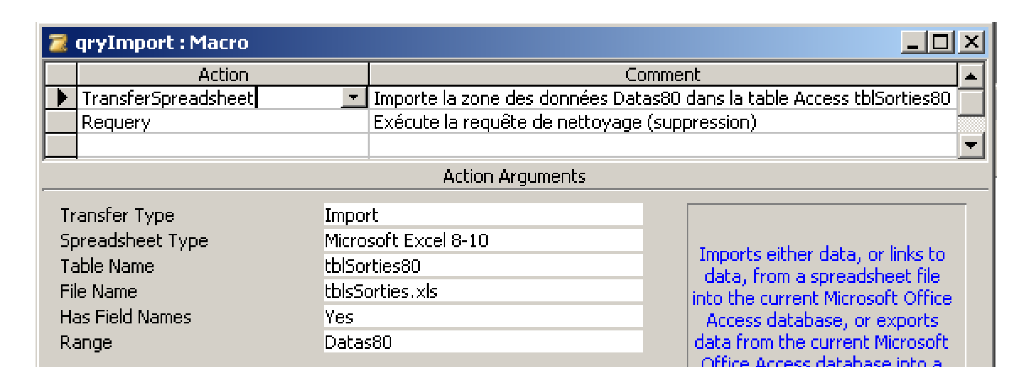
Qu'observez-vous ? Quelles sont les contraintes

La table 1980 (un import et nettoyage par une macro incluant une requête de suppression:

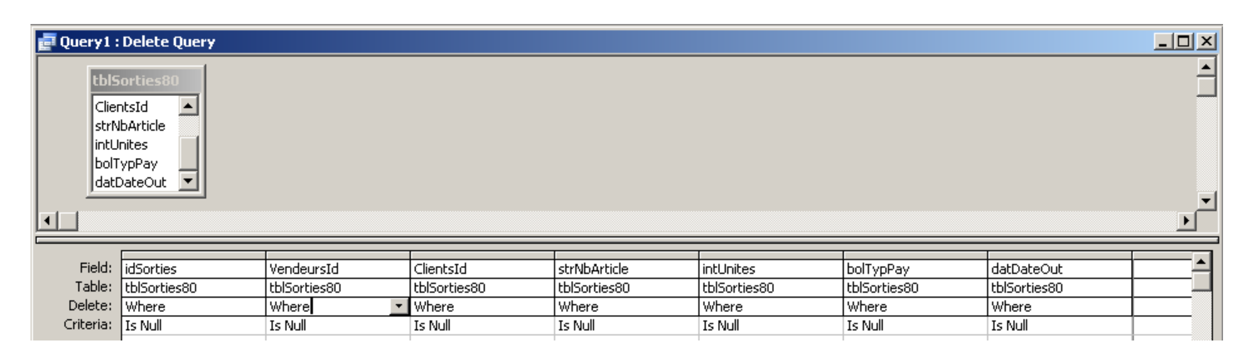
tblSorties80 : Table

idSorties	VendeursId	ClientsId	strNbArticle	intUnites	bolTypPay	datDateOut
1	5	3	INF-003	24	-1	28.11.1980
3	8	2	GEN-001	43	0	10.12.1980
2	11	2	GEN-003	43	0	10.12.1980

La macro correspondante:



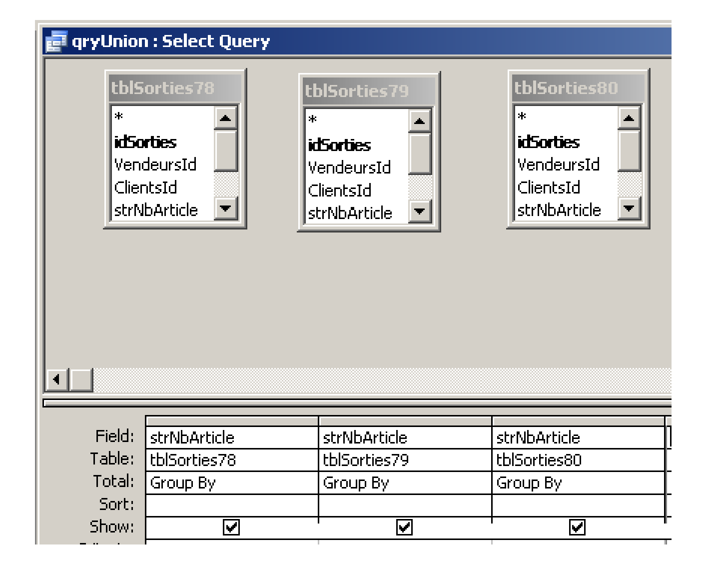
et la requête de suppression correspondante:



L'objectif maintenant est de faire de ces trois tables importées (qui habituellement ne viennent pas de MS Excel mais sont créés et conservées dans MS Access) une unique table qui indique

quels sont les produits qui ont été vendus en entre 1978 et 1980 (remarquez bien que certains articles se répètent !).

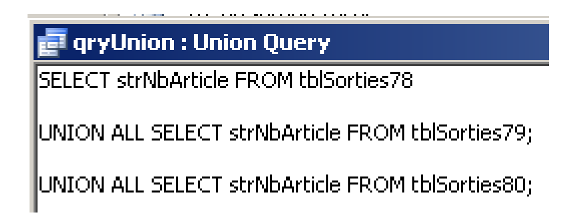
Si vous créez une requête normale (c'est celle que les gens choisissent souvent en premier):



Vous n'arriverez à rien ! Bien sûr, vous pouvez faire une multitude de "Requêtes d'ajout" mais lorsque les données sont nombreuses, la méthode est alors catastrophique!!! Bon ceci dit cela reste un bon exercice et c'est quand même le cas le plus courant qui évite d'apprendre par coeur le langage SQL. Vous pouvez donc aussi le faire en tant qu'exercice!

La solution est alors la suivante:

Il faut créer une requête en mode création et passer directement en affichage SQL et saisir:

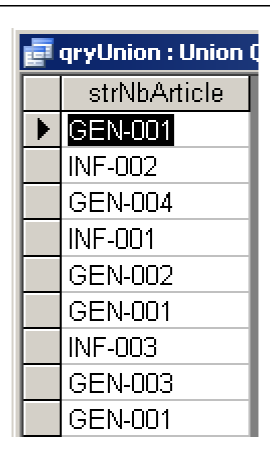


et vous obtenez comme symbole pour cette requête dans la fenêtre de base de données:



d'où le nom de "requête d'union".

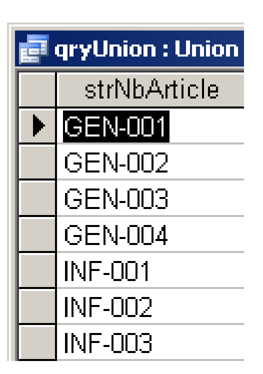
En exécutant cette requête, vous obtenez:



qryUnion : Union Q

strNbArticle
GEN-001
INF-002
GEN-004
INF-001
GEN-002
GEN-001
INF-003
GEN-003
GEN-001

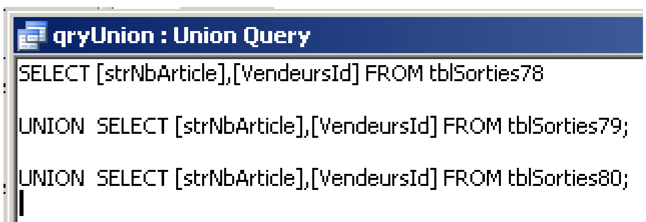
Nous avons obtenu ce que nous voulions mais… avec les doublons en plus. Pour les éliminer, il faut enlever les "ALL" de la requête SQL. Il vient alors:



qryUnion : Union

strNbArticle
GEN-001
GEN-002
GEN-003
GEN-004
INF-001
INF-002
INF-003

Si nous rajoutons des paramètres pour les résultats d'affichage, nous voyons que nous passons alors de 7 enregistrements à 9 (pas besoin de mettre les crochets ! c'est juste pour montrer une autre façon d'écrire):



qryUnion : Union Query

```
SELECT [strNbArticle],[VendeursId] FROM tblSorties78

UNION  SELECT [strNbArticle],[VendeursId] FROM tblSorties79;

UNION  SELECT [strNbArticle],[VendeursId] FROM tblSorties80;
```

Figure 8 Requête d'union

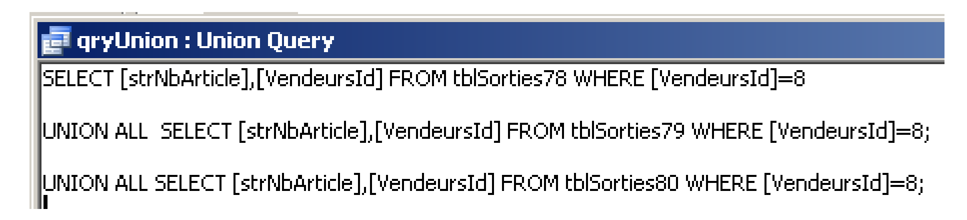
qryUnion : Union Query

strNbArticle	VendeursId
GEN-001	1
GEN-001	4
GEN-001	8
GEN-002	2
GEN-003	11
GEN-004	8
INF-001	7
INF-002	3
INF-003	5

Ce résultat est totalement normal car le couple (Article,Vendeurs) devient maintenant unique. En l'occurrence les GEN-001.

Remarque: Vous pouvez changer le "Union" par un "Intersect" ou un "Minus" sur d'autres systèmes de base de données que MS Access… désolé…

Ensuite, rien ne vous empêche d'ajouter des critères en spécifiant la commande "WHERE", tel que par exemple (nous voulons que le vendeur numéro 8 !):



Exercice: Dans la requête précédente, faites que les articles soient triés dans l'ordre décroissant de leurs noms.

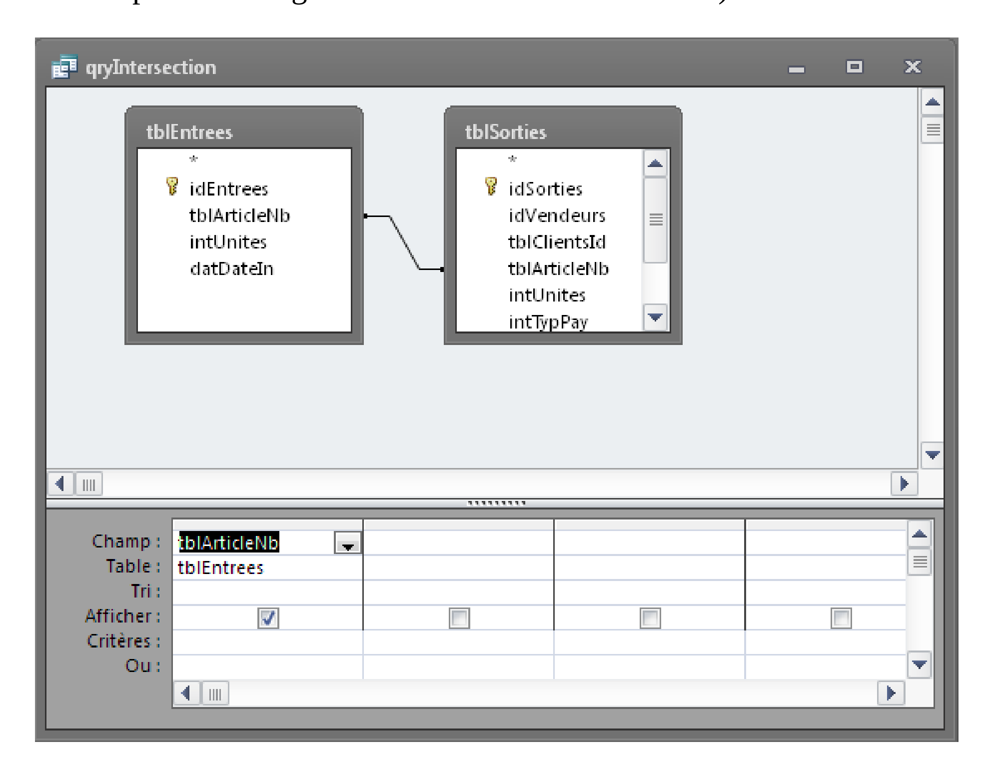
Remarque: nous allons voir de suite une autre application très concrète de ces requêtes d'union.

7.19 Requête d'intersection

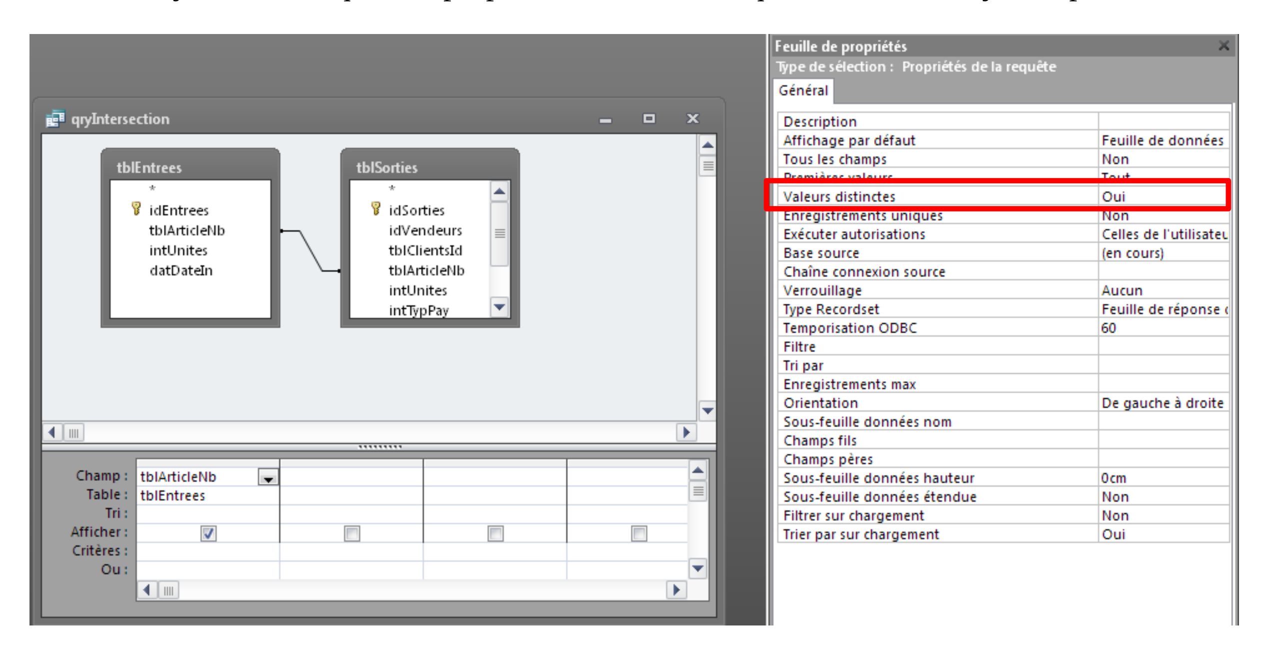
La requête d'intersection permet par exemple de sortir que les champs de deux tables qui existent respectivement dans l'une et dans l'autre.

Pour exemple, nous voulons la liste des articles qui sont entrés et sortis! En d'autres termes: parmi la liste des articles qui ont été notés dans les entrées de stocks, ceux qui ont déjà eu une sortie.

Comme dans MS Access il n'existe pas encore la fonction SQL qui se nomme INTERSECT, il d'abord faut créer des relations entre les champs qui doivent être identiques entre les deux tables (dans le cas présent il s'agit seulement du nom des articles):



Et il faut rajouter à la requête la propriété de distinction que nous avons déjà vue plus haut:



En d'autres termes (pour parler en SQL), on fait donc un DISTINCT d'un INNER JOIN

7.20 Requête de comptage

Considérons la table suivante:

tblSorties : Table

idSorties	Vendeurs	Client	Code Article	Unités vendues	Type de paieme
1	Massa	Boltzmann	GEN-001	50	-1
2	Castrini	Angel	GEN-003	10	0
4	Clerc	Dutron	INF-002	1500	0
5	Butty	Dutron	GEN-002	100	-1
6	Mettraux	Angel	GEN-003	20	0
8	Massa	Boltzmann	INF-002	500	0
9	Hoffmann	Dutron	INF-003	30	0
10	Clerc	Boltzmann	GEN-002	50	0
11	Massa	Angel	INF-001	5	0
12	Mettrez	Angel	INF-003	10	0
13		Angel	GEN-004	80	0
14	Butty	Dutron	INF-001	15	0
15	Clerc	Boltzmann	INF-003	5	0
16	Castrini	Dutron	GEN-003	30	0
17	Weidman	Angel	GEN-006	200	0
18	Mettraux	Dutron	INF-002	750	0
19	Hoffmann	Angel	GEN-001	100	0
20	Castrini	Angel	GEN-004	30	0
21	Clerc	Boltzmann	INF-002	100	0
22	Massa	Boltzmann	GEN-003	30	0
23	Mettrez	Boltzmann	INF-002	20	0
24	Butty	Dutron	INF-004	750	0

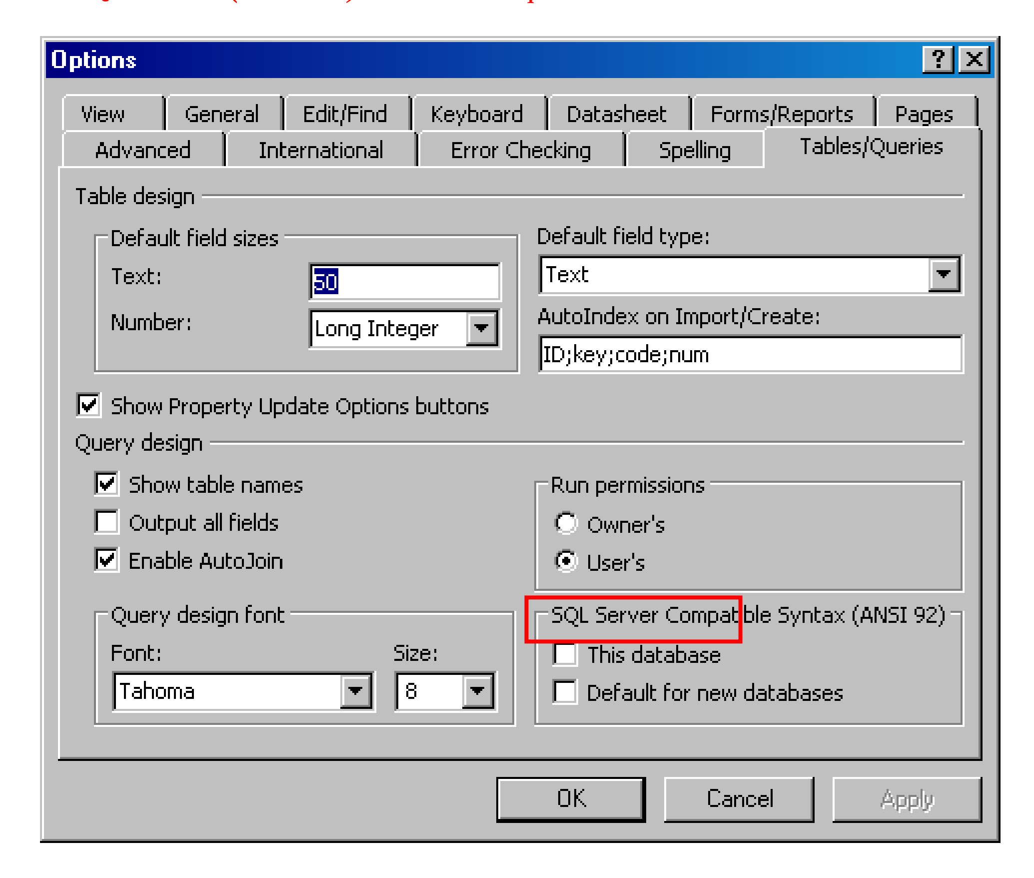
L'idée est de compter combien de clients (uniques) ont fait une commande… malheureusement la solution n'est pas simple et faire un simple regroupement par nom de client ne suffit pas. Nous donnons alors la solution (ne connaissant pas plus simple à ce jour):

Requête1 : Requête Sélection

```
SELECT Count(*) AS [NbrCustomers] FROM (select distinct [ClientsId]
from tblSorties)
```

Nommez cette requête *qrySQLCount*.

Remarque: Certains utilisateurs font usage de MS Access pour tester des commandes SQL avant de les exécuter dans SQL Server. Si c'est votre cas, n'oubliez pas d'aller dans le menu *Outils/Options* de MS Access et dans l'onglet *Tables/Requêtes* d'activer l'option *Syntaxe Compatible SQL Server (ANSI 92)* comme indiqué ci-dessous:



MS Access utilisant par défaut le ANSI 89… ce qui date un peu quand même…

Attention à bien faire une copie de la base de données avant l'activation de cette option comme vous l'indiquera le système!

7.21 Requête d'union (bis)

Imaginons que nous souhaiterions une table (qui servira pour la construction d'un graphique) avec le nombre de clients qui ont acheté quelque chose avec le nombre de clients total.

Pour cela il nous faudra d'abord la requête faite précédemment mais renommée *qrySQLCountOrd*:

Requête1 : Requête Sélection

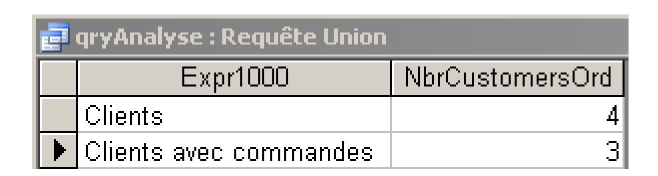
```
SELECT Count(*) AS [NbrCustomers] FROM (select distinct [ClientsId]
from tblSorties)
```

et une requête similaire (que nous laisserons le soin au participant de créer) qui comptera simplement le nombre de clients à partir de la table *tblClients* et que le participant nommera *qrySQLCountCust*.

Nous allons maintenant créer la requête d'union suivante:



Qui donne le résultat suivant dans le cadre de notre petite base de données:



qryAnalyse : Requête Union

Expr1000	NbrCustomersOrd
Clients	4
Clients avec commandes	3

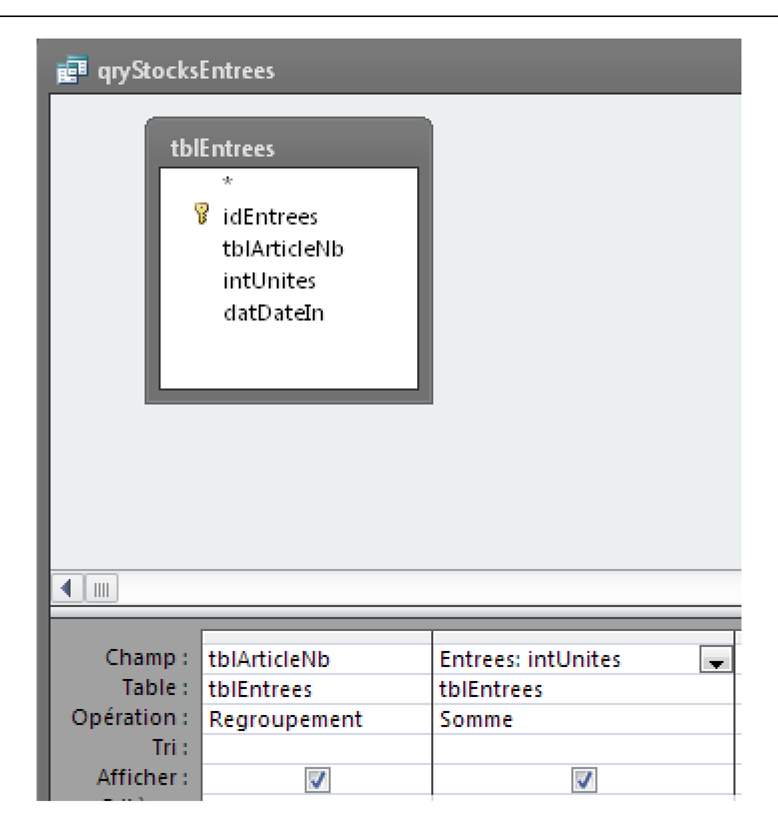
Essayez comme exercice de compiler les trois requêtes en une seule uniquement avec code SQL.

7.22 Requêtes de requêtes

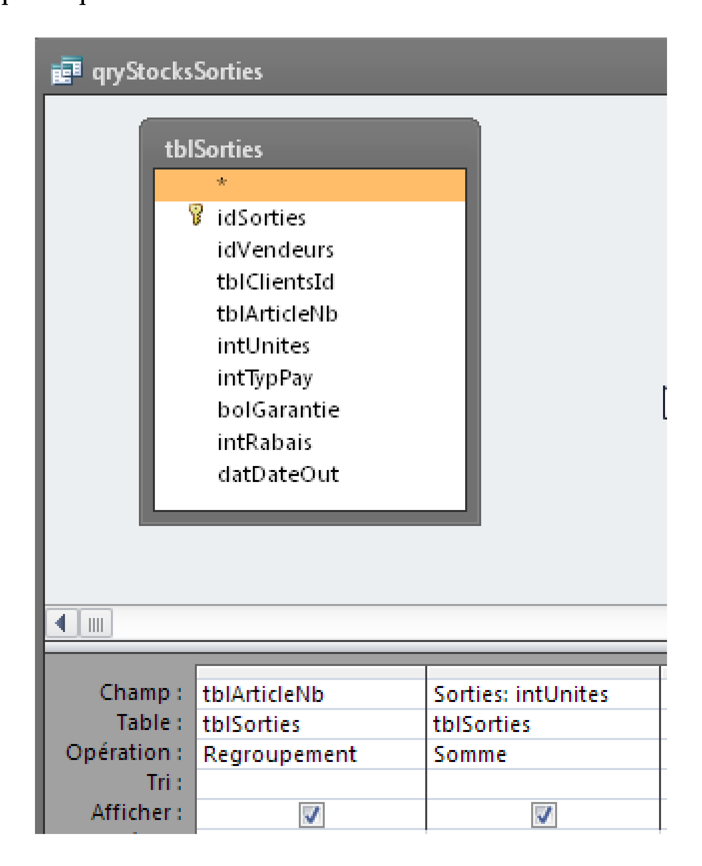
Créons plusieurs requêtes (en s'interdisant de faire du SQL) nommées respectivement *qryStocksEntrees*, *qryStocksSorties* et *qryStocks* qui calcule les stocks (différence entre vendus et achetés) du magasin et ce pour chaque article:

Nous allons voir qu'il va être nécessaire de faire une requête de requêtes pour arriver au résultat.

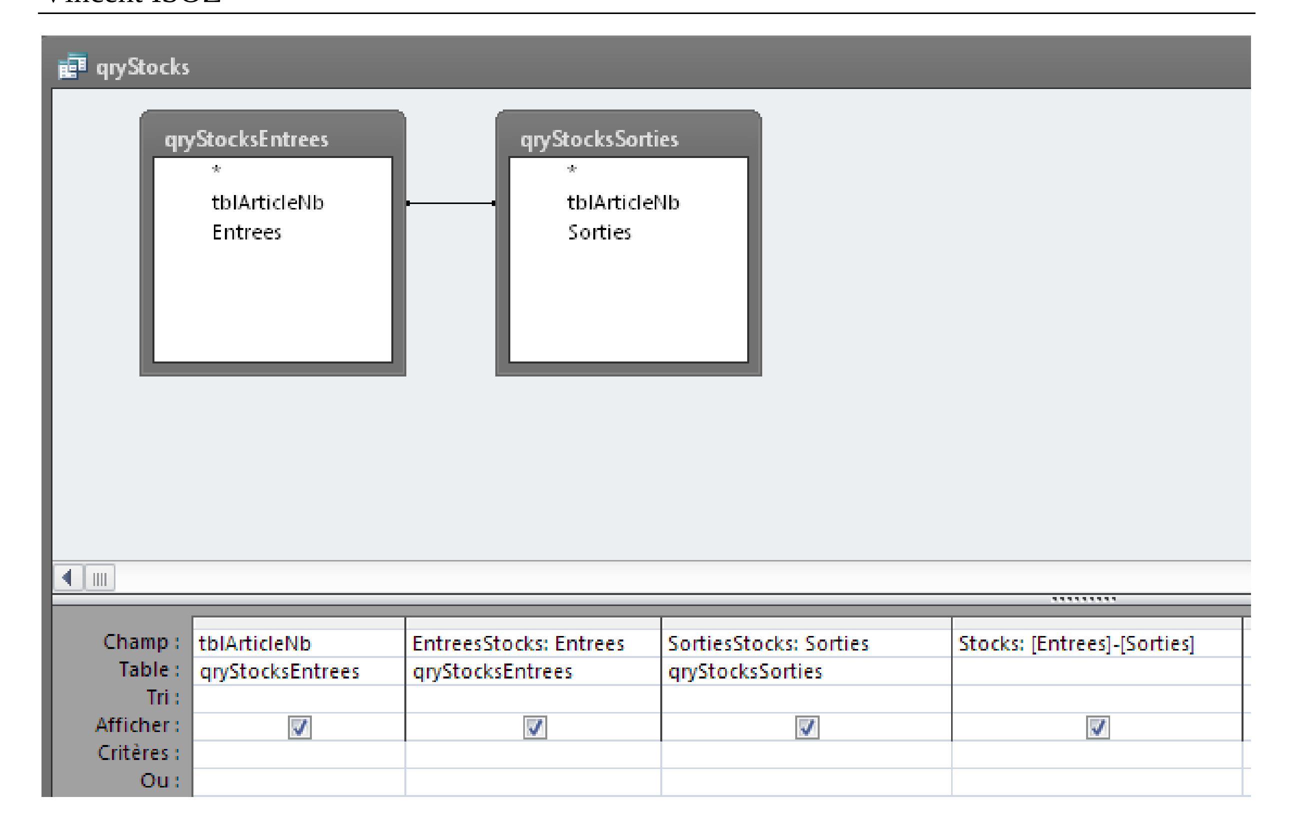
Nous construisons d'abord une requête qui résume les entrées:



Ensuite la même requête qui résume les sorties:



et enfin la requête qui utilise les deux précédentes:



7.23 Divers (exercices)

Exercice: Dans la même base, créer une première requête *qryQuiQuoi* qui affiche dans une table, quel vendeur a vendu quoi et à qui (client) et une deuxième requête *qryQuoiQui* qui affiche quel client a acheté quoi avec l'aide de quel vendeur.

8 États (Rapport)-Formulaires (complexes)

On peut imprimer les tables dans MS Access pour consultation et vérification. Cependant, il est peut être préférable avant d'envoyer à l'imprimante une table avec 10'000 enregistrements, de vérifier la façon dont l'impression va être effectuée.

L'objectif des états (rapports/reporting) est trivialement:

- De déterminer quelles sont les informations vraiment critiques au fonctionnement de l'entreprise et le coût de ces informations relativement à ce qu'elles permettent d'économiser
- Avoir un compte rendu de l'état de lieux de certaines informations filtrées / triées / regroupées ou non et ce de façon visuelle et synthétique ou non (mise en forme, graphique, etc.)
- De réaliser divers calculs (totaux, moyennes, écarts, comparatif d'une période à l'autre, etc.)
- De support d'aide à la décision

Remarque: L'informatique décisionnelle (en anglais : DSS pourDecision Support System ou encore BI pour Business Intelligence) désigne les moyens, les outils et les méthodes qui permettent de collecter, consolider, modéliser et restituer les données, matérielles ou immatérielles, d'une entreprise en vue d'offrir une aide à la décision et de permettre aux responsables de la stratégie d'entreprise d’avoir une vue d’ensemble de l'activité traitée.

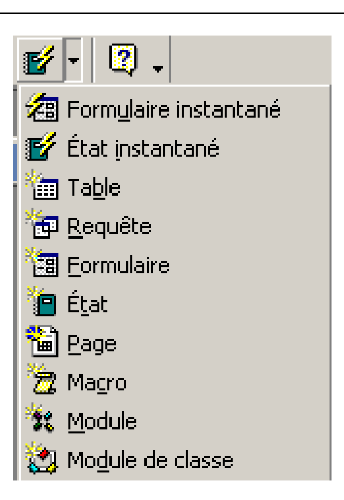
Pour cela, avant d'imprimer quoi que ce soit, comme dans tout autre logiciel, il faut prendre l'habitude d'activer l'Aperçu avant impression et au besoin de définir ses propres propriétés d'impression qui sont peu nombreuses...

Par exemple, ouvrez la table *Articles* et activez l'aperçu avant impression et définissez une mise en page "paysage".

Cependant, on utilisera plus fréquemment les états pour effectuer une impression professionnelle des données de la base. On verra plus tard comment personnaliser ses états avec tous les soins nécessaires.

Par exemple, sélectionnez la requête "Cible Fournisseurs" et créez en un état à l'aide de l'assistant (du type de celui visible à la page suivante). Vous enregistrerez cet état sous le nom *rptCibleFournisseurs*.

Pour créer un état instantané, ouvrir une table et:



Evidemment, il faut rendre l'état accessible à l'utilisateur de votre base à partir d'un bouton se situant sur un formulaire si l'on veut bien faire les choses.

Maintenant que nous avons parcouru les principaux objets d'Access (Tables, filtres, requêtes, formulaires (sur requêtes et sur tables), états) vous pensez nécessairement qu'il est dommage que l'on doive passer à chaque fois par la fenêtre de base de données pour activer ces derniers?!

Il n'est est rien! Au cours "avancé" (ou lors de la dernière demi-journée), nous verrons comment créer une belle interface utilisateur avec tous les boutons (il y a du code VBA derrière) nécessaire à une utilisation optimale de votre base. Il en sera de même pour le choix de l'ouverture automatique d'un des formulaires, lors de l'ouverture de la base.

Vous avez ci-dessous un état créé avec l'assistant qui groupe les articles de la table *tblSorties* par ordre alphabétique et qui indique les informations de sortie des articles (Vendeurs, Unités vendues, Type de payement, Date de sortie) et qui pour chaque article fait une synthèse des informations (Somme du nombre d'articles et pourcentage par rapport à la totalité).

Sorties

txt_nbarticle GEN-001

Vendeurs	Unités vendues	Payé par crédit	Date sortie
Weidman	50	☑	28.11.1996
Hoffmann	100	☐	08.01.1997

Summary for 'txt_nbarticle' = GEN-001 (2 detail records)
Sum 150)ui
Percent 3.38% 7%

txt_nbarticle GEN-002

Vendeurs	Unités vendues	Payé par crédit	Date sortie
Butty	100	☑	11.12.1996
Clerc	50	☐	18.12.1996

Summary for 'txt_nbarticle' = GEN-002 (2 detail records)
Sum 150)ui
Percent 3.38% 7%

txt_nbarticle GEN-003

Vendeurs	Unités vendues	Payé par crédit	Date sortie
Castrini	10	☐	10.12.1996
Castrini	30	☐	02.01.1997

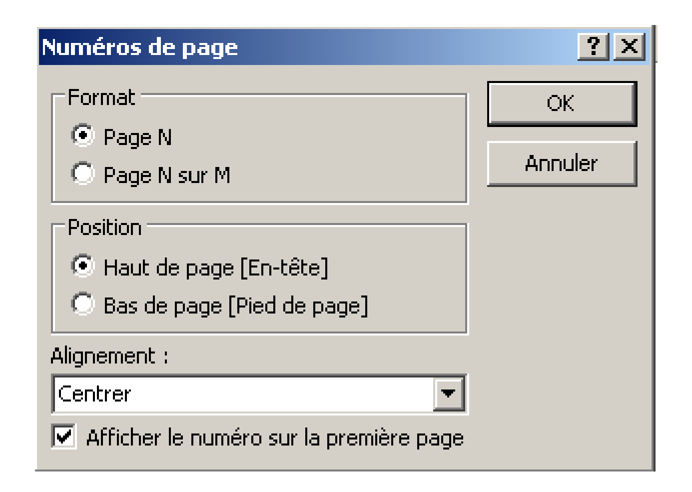
Il existe également au besoin une option qui permet d'exporter les données représentées par l'état vers un fichier Word ou Excel. Le système de marche pas toujours comme on le souhaite et n'est pas toujours fonctionnel (dépendant de la façon avec laquelle le logiciel a été installé sur la machine et de la complexité de la base de données en question).



Comme exercice il vous est maintenant demandé de créer un état que vous nommerez *rptLettreType* qui génère automatique une lettre type de bienvenue à chaque nouveau client inscrit à notre système fidélité du magasin.

La lettre type (outre le logo que vous pouvez choisir) doit contenir les informations standards du client tel que l'expéditeur, l'adresse de son destinataire, les formules de politesses adéquatement conjuguées, etc.

Si par mégarde, la numérotation des pages des rapports est supprimée, il est toujours possible de la réinsérer à un endroit quelconque en allant dans le menu *Insertion/Numéros de pages*. Apparaît alors à l'écran l'option suivante:

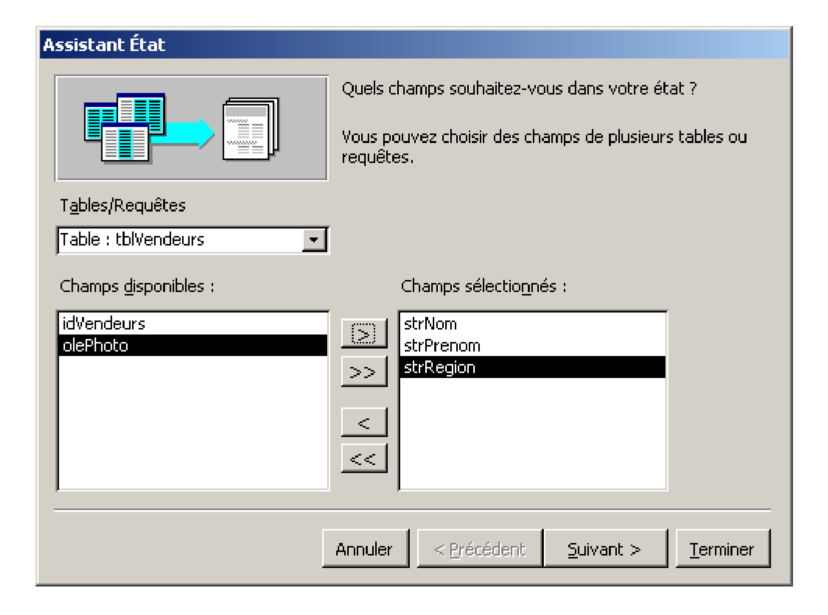


Il suffit de choisir et de valider par OK.

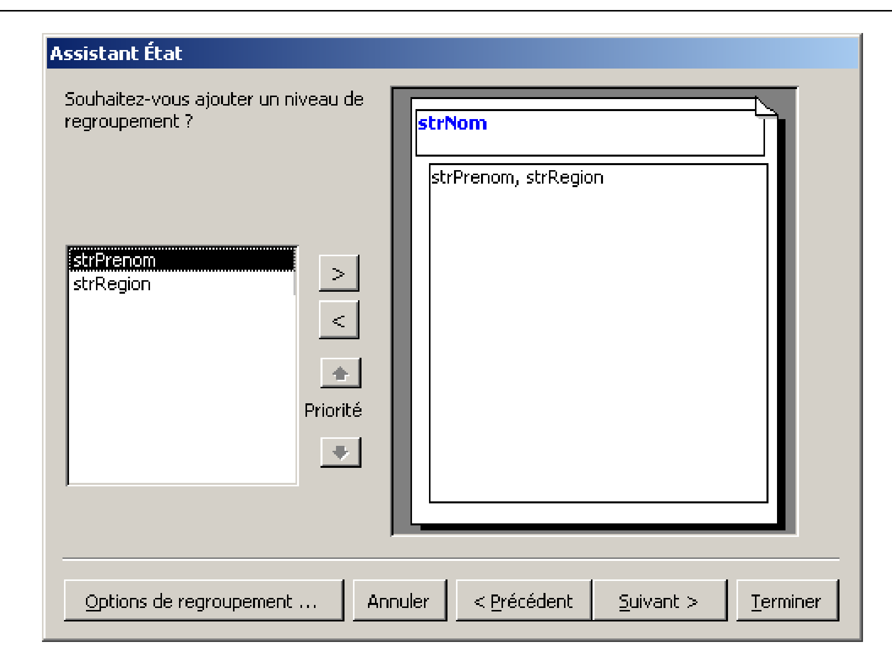
8.1 Carnet d'adresse

Nous allons voir ici comment créer, toujours avec l'aide de l'assistant, un petit carnet d'adresse des vendeurs.

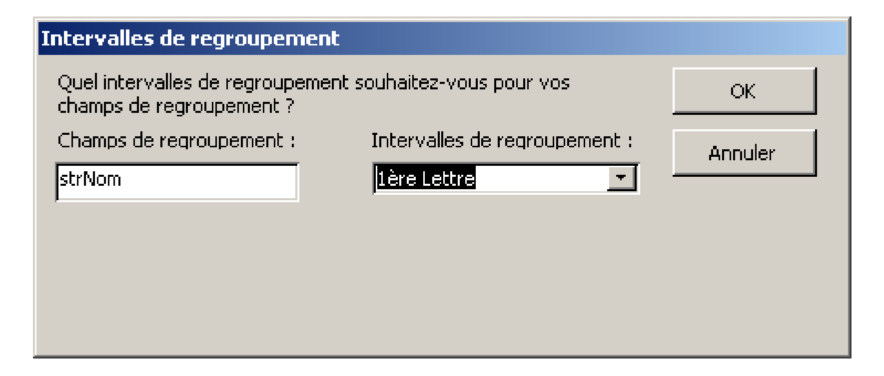
Lancez pour cela l'assistant de création de rapport en choisissant la table *tlbVendeur*:



Cliquez sur *Suivant* et ensuite choisissez de faire un regroupement par nom:



ensuite, cliquez sur Options de regroupement en choisissez:



Validez enfin par Terminer et observez le résultat (dont l'esthétique est toujours discutable et demande un peu de travail de la part du développeur):

tblVendeurs

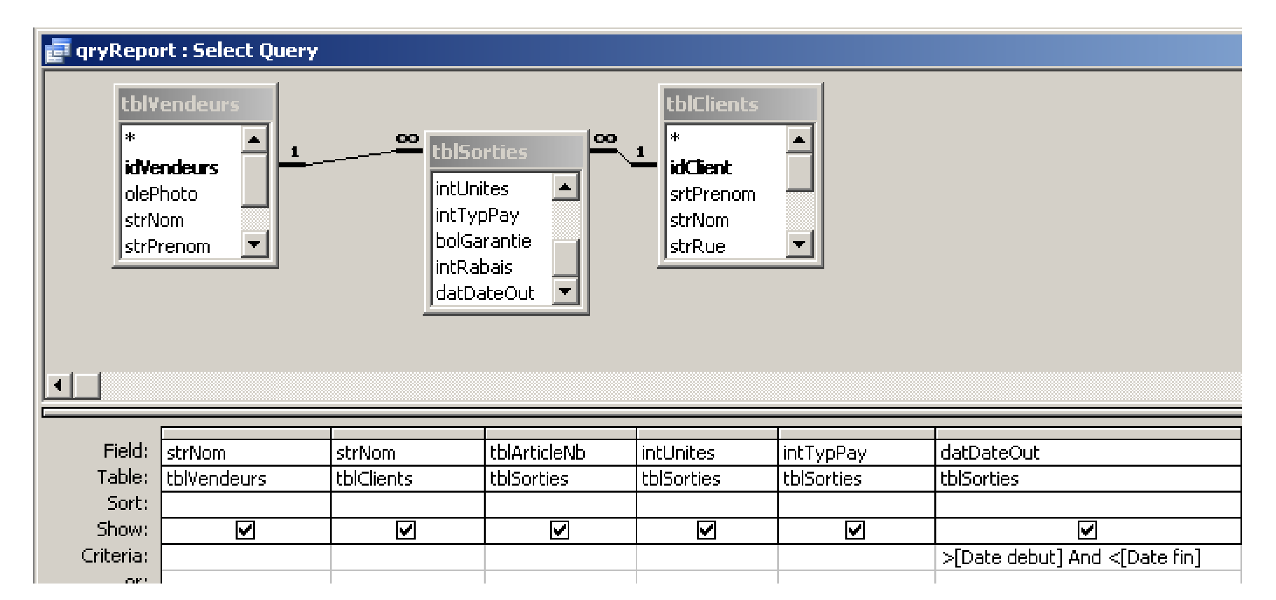
strNom par 1ère Lettre	*Nom*	*Prénom*	*Région*
B			
	BARBIER	Aline	Nord
	BUTTY	Joe	Sud
C			
	CASTRINI	Rodolfo	Nord
	CLERC	Marlène	Nord
D			
	DE SIEBENTAL	Willi	Est
H			
	HOFFMANN	Thérèse	Ouest
M			
	MASSA	Sybille	Ouest
	METTRAUX	Théo	Est
	METTREZ	Sébastien	Sud
	MOUTER	Jean	Nord

8.2 Synthèse et requête

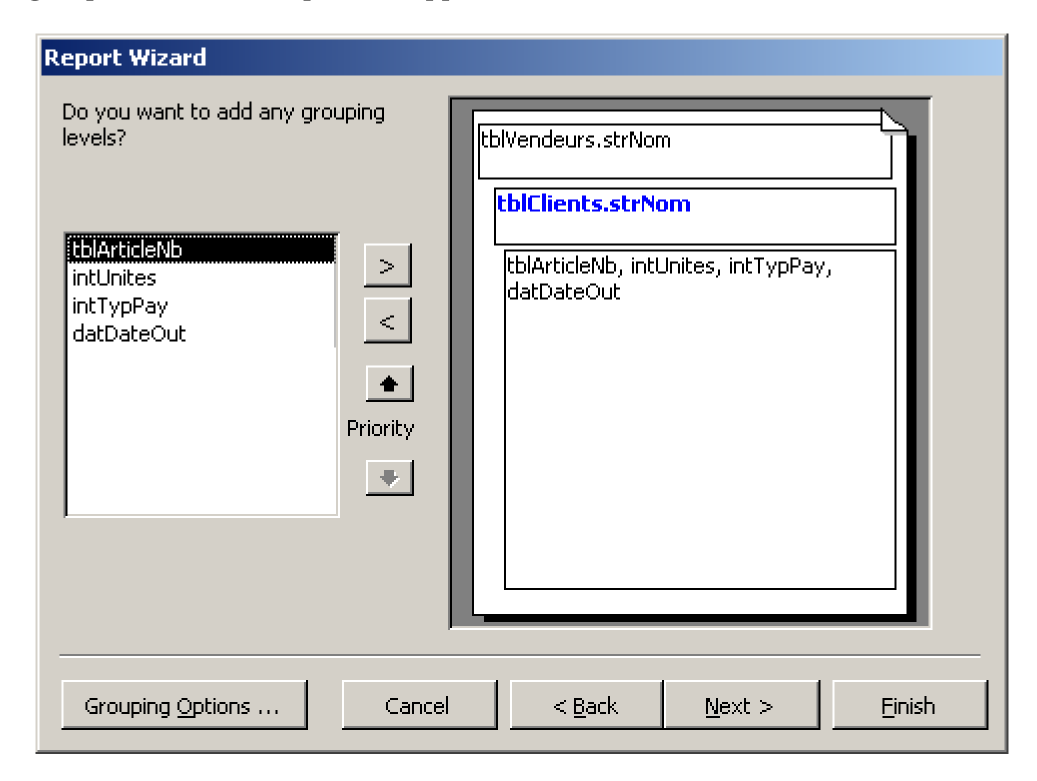
Créez un état (*rptSorties*) basé sur une requête paramétrée qui affiche toutes les ventes de tous les vendeurs (les ventes doivent êtres groupées par vendeur et par date croissante) entre deux dates que l'utilisateur doit pouvoir choisir dans un formulaire.

Remarque: le design du formulaire n'est pas important (ce n'est pas un cours pour pigistes)

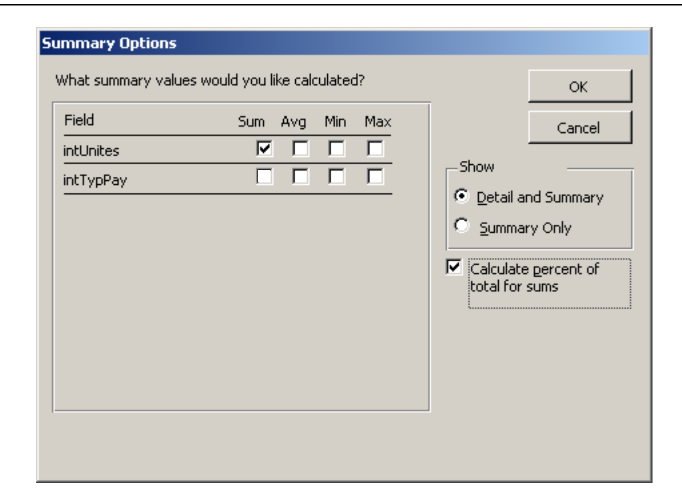
Voici grosso modo la requête (nommée *qryReport*):



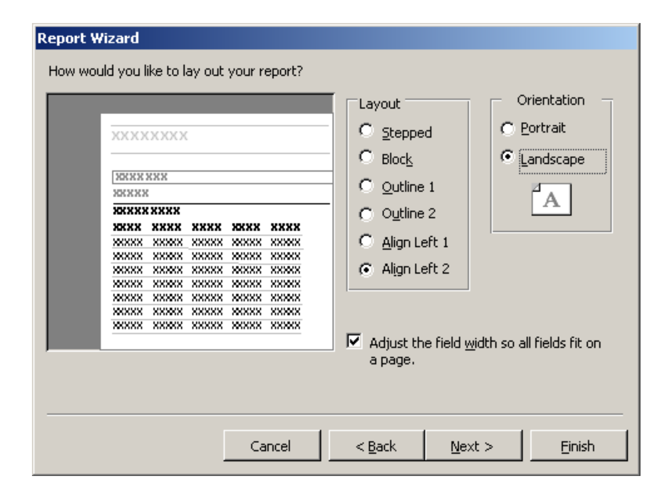
Voici le groupement souhaité pour le rapport:



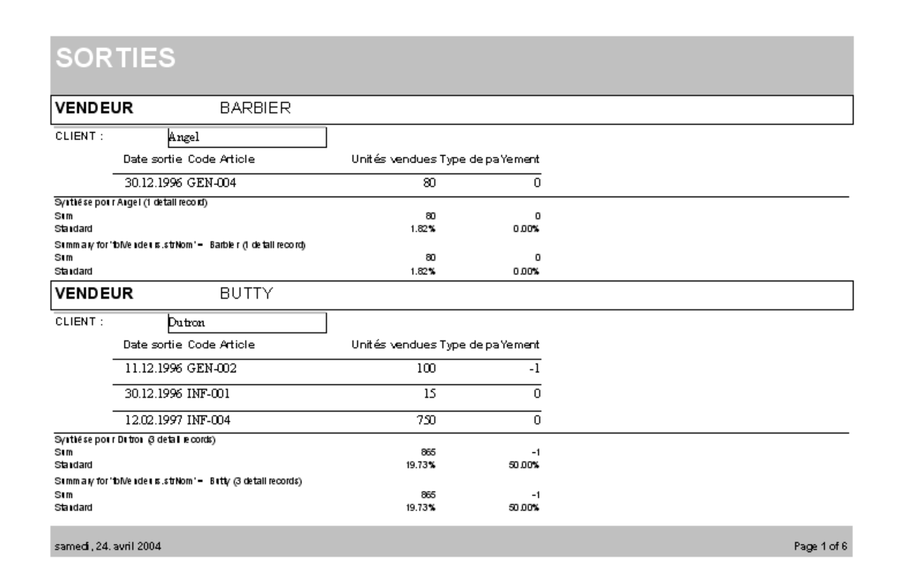
et les options de synthèse:



Le type de mise en page:



et une idée du rapport correspondant (c'est moche mais bon…):



SORTIES

VENDEUR BARBIER

CLIENT : Angel

Date sortie	Code Article	Unités vendues	Type de paYement
30.12.1996	GEN-004	80	0

Synthèse pour Angel (1 detail record)

Sum	80	0
Standard	1.82%	0.00%

Summary for 'tblVendeurs.strNom' = Barbier (1 detail record)

Sum	80	0
Standard	1.82%	0.00%

VENDEUR BUTTY

CLIENT : Dutron

Date sortie	Code Article	Unités vendues	Type de paYement
11.12.1996	GEN-002	100	-1
30.12.1996	INF-001	15	0
12.02.1997	INF-004	750	0

Synthèse pour Dutron (3 detail records)

Sum	865	-1
Standard	19.73%	50.00%

Summary for 'tblVendeurs.strNom' = Butty (3 detail records)

Sum	865	-1
Standard	19.73%	50.00%

samedi, 24. avril 2004 Page 1 of 6

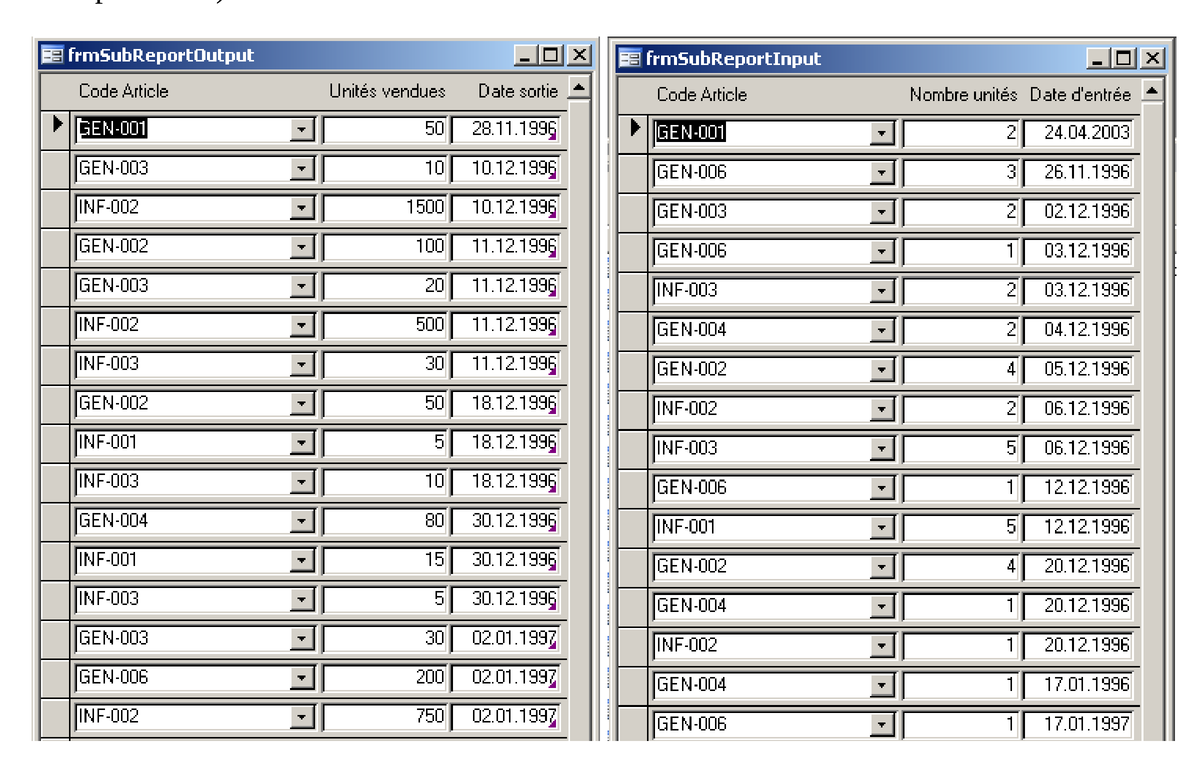
Personnalisez cet état maintenant avec votre formateur (rajout de formules en pied et tête de page ainsi qu'en pied de rapport, mise en forme conditionnelle, DLookUp,…).

8.3 Sous-états

Nous allons dans cet exemple créer un étant avec des sous-états (ou sous-formulaires pour être plus exact). L'objectif sera d'avoir un rapport qui nous indique pour chaque article, une colonne avec les sorties et une colonne avec les entrées.

L'esthétique du résultat ne sera pas prise en compte car seule nous intéresse la technique et la possibilité offerte par MS Access (le formatage étant à ce niveau supposé maîtrisé).

Pour cet exemple, il va nous falloir dans un premier temps créer deux formulaires respectivement pour les tables *tblEntrees* et *tblSorties* ayant le design et les noms visible dans les capture d'écran ci-dessous (à ce niveau du cours, créer ce genre de formulaires ne doit plus être un problème):



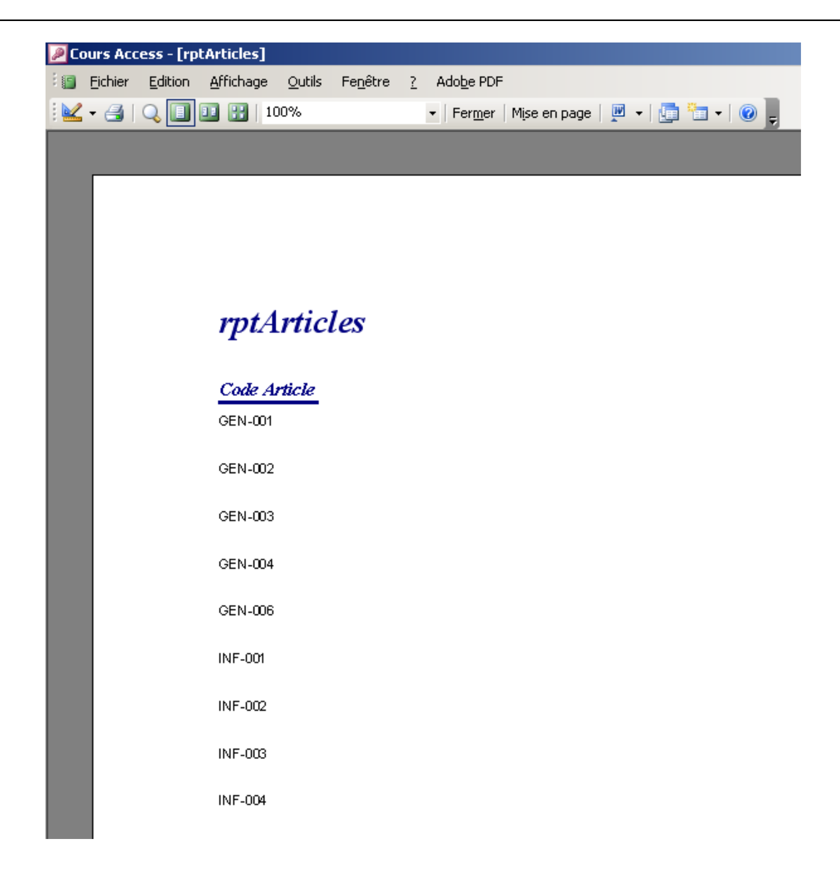
frmSubReportOutput

Code Article	Unités vendues	Date sortie
GEN-001	50	28.11.1996
GEN-003	10	10.12.1996
INF-002	1500	10.12.1996
GEN-002	100	11.12.1996
GEN-003	20	11.12.1996
INF-002	500	11.12.1996
INF-003	30	11.12.1996
GEN-002	50	18.12.1996
INF-001	5	18.12.1996
INF-003	10	18.12.1996
GEN-004	80	30.12.1996
INF-001	15	30.12.1996
INF-003	5	30.12.1996
GEN-003	30	02.01.1997
GEN-006	200	02.01.1997
INF-002	750	02.01.1997

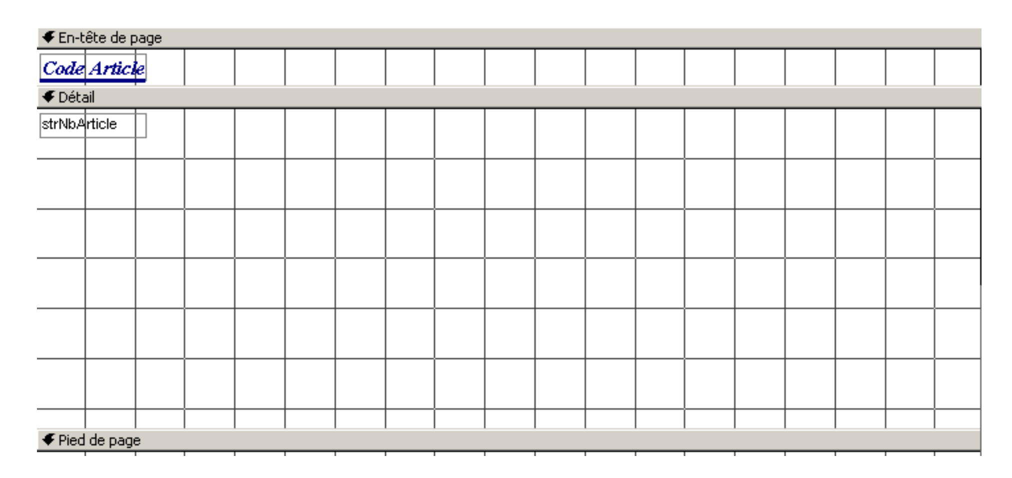
frmSubReportInput

Code Article	Nombre unités	Date d'entrée
GEN-001	2	24.04.2003
GEN-006	3	26.11.1996
GEN-003	2	02.12.1996
GEN-006	1	03.12.1996
INF-003	2	03.12.1996
GEN-004	2	04.12.1996
GEN-002	4	05.12.1996
INF-002	2	06.12.1996
INF-003	5	06.12.1996
GEN-006	1	12.12.1996
INF-001	5	12.12.1996
GEN-002	4	20.12.1996
GEN-004	1	20.12.1996
INF-002	1	20.12.1996
GEN-004	1	17.01.1996
GEN-006	1	17.01.1997

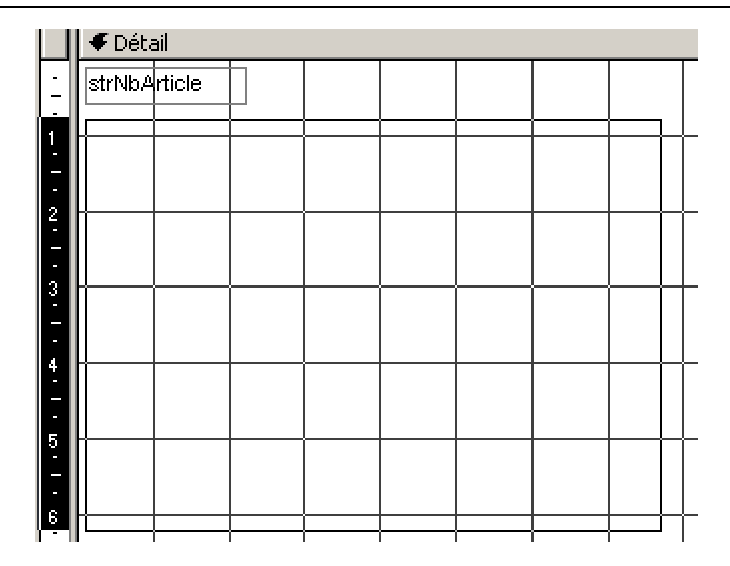
Nous aurons également besoin d'un rapport basé sur la table *tblArticles* et nommé *rptArticles* affichant simplement la chose suivante:



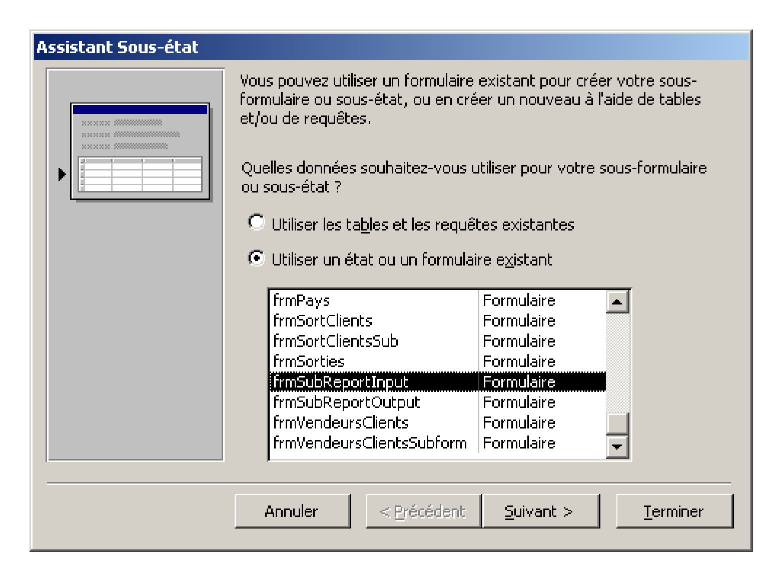
Maintenant, nous allons passer cet état en mode création et élargie la zone *Détails*:



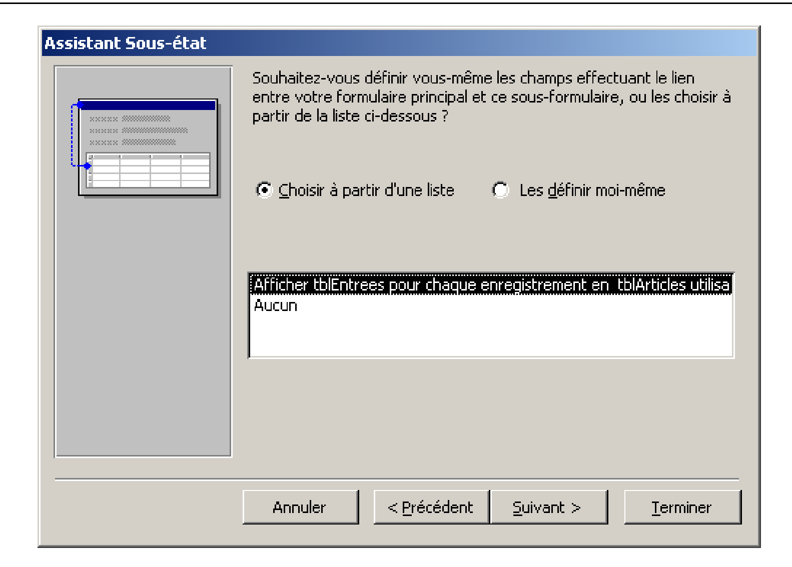
Ensuite, dans la boîte à outils contrôles, nous cliquons sur le bouton *Sous-formulaire/Sous-état* et nous traçons une zone d'une taille choisie au bolomètre pour commencer que nous pourrons toujours changer plus tard:



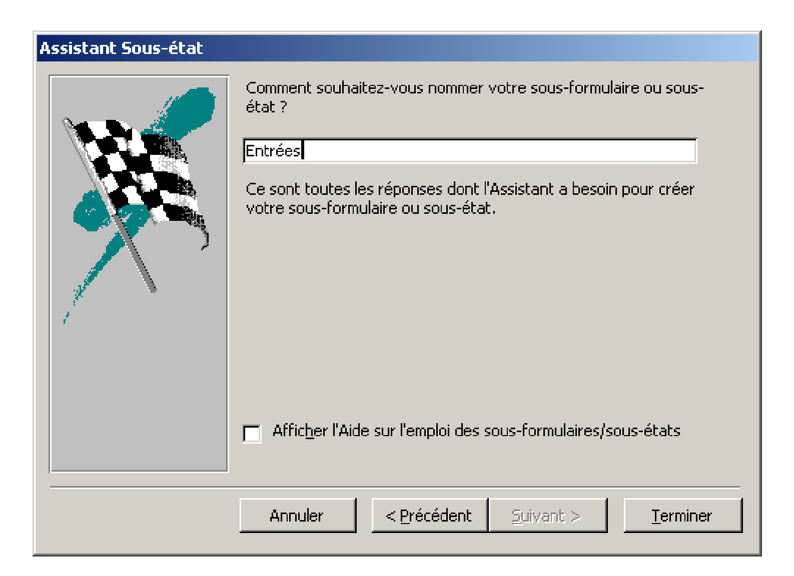
Apparaît alors l'assistant suivant dans lequel il vous est demandé pour l'exemple de sélectionner un des deux formulaires créés précédemment:



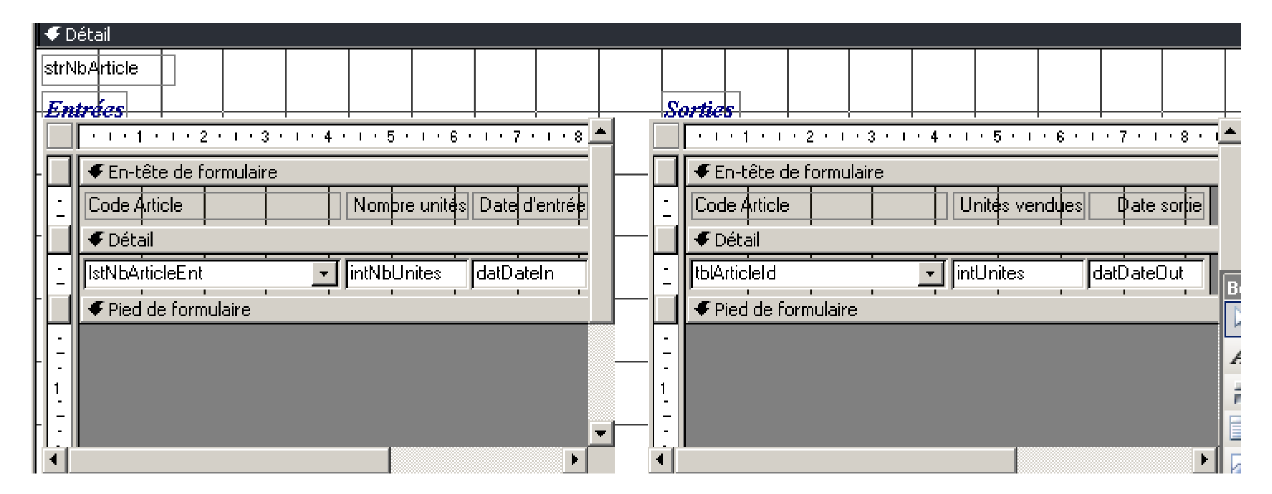
Cliquez sur *Suivant* et dans la fenêtre suivante ne touchez à rien. Cliquez simplement sur *Suivant* (MS Access s'occupe de faire les liaisons qu'il faut tout seul comme un grand dans un cas si trivial):



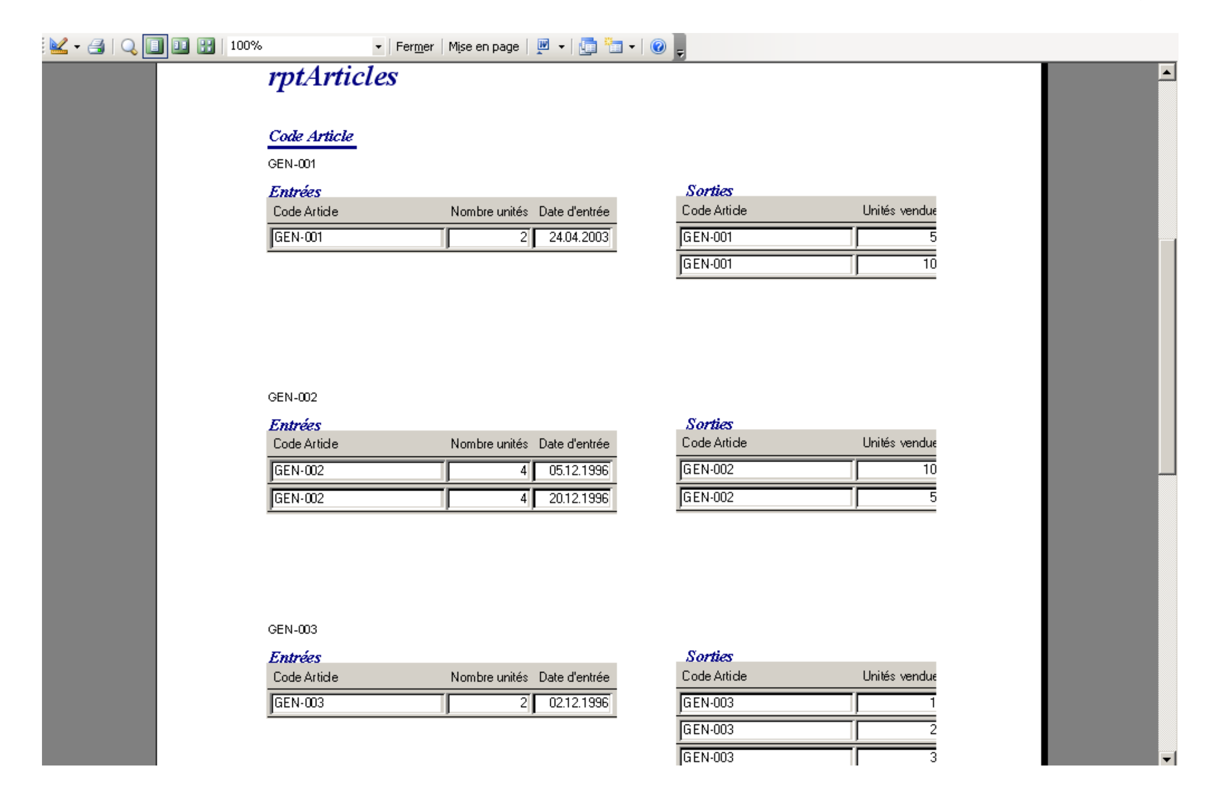
Cliquez sur suivant et saisissez ce qui convient:



Validez par *Terminer*. Faites de même avec le deuxième sous-formulaire de manière à obtenir quelque chose du genre (vous pouvez prendre l'incitative d'améliorer l'esthétique si le cœur vous en dit):



Le rapport donne alors (normalement on prendra soin de supprimer l'information à double dans le sous-formulaire affichant une deuxième fois le code de l'article bien évidemment):



8.4 Rapports avec groupes (pour lettres ou factures)

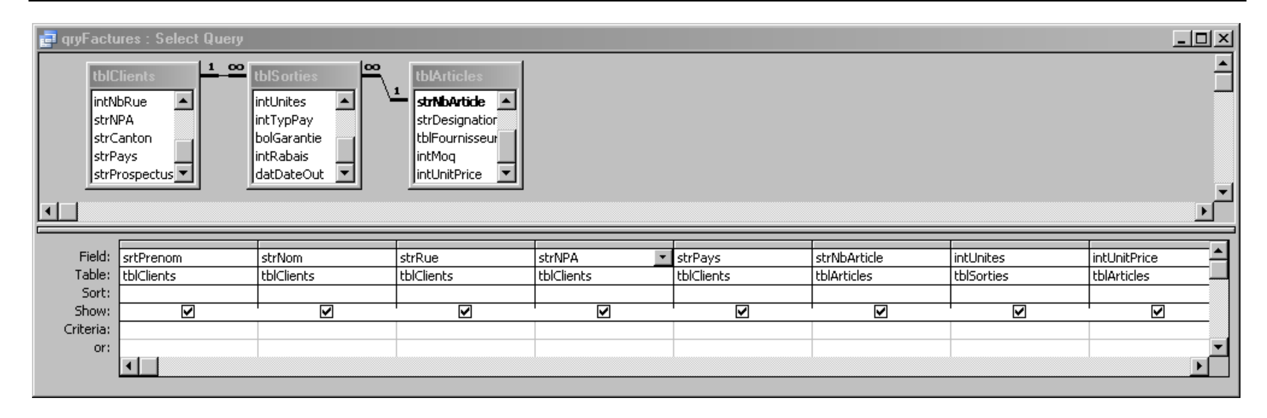
Dans de nombreuses entreprises il est demandé d'avoir des états Access avec un en-tête et pied de page corporate avec zone d'en-tête de document comprenant adresses du destinataire et de l'expéditeur ainsi qu'un numéro de référence et un texte d'accompagnement standard.

En-dessous de l'en-tête de document il est systématique que les entreprise souhaitent un tableau avec un listing venant d'une requête Access représentant soit les détails d'une facture, soit les détails de statistiques diverses.

Il n'est pas possible d'utiliser la technique des sous-rapport vue précédemment pour cela car… les entreprises ont souvent sous le listing un autre texte d'accompagnement de fin de document ou des statistiques qui doivent se placer automatiquement en-dessous de la table du milieu du document et ce quel que soit sa taille (qui est donc variable).

Il faut alors utiliser une technique utilisant le concept de **Groupes** avec des requêtes comprenant absolument toutes les informations pour la lettre.

Pour cela, créons d'abord la requête simplifiée suivante:



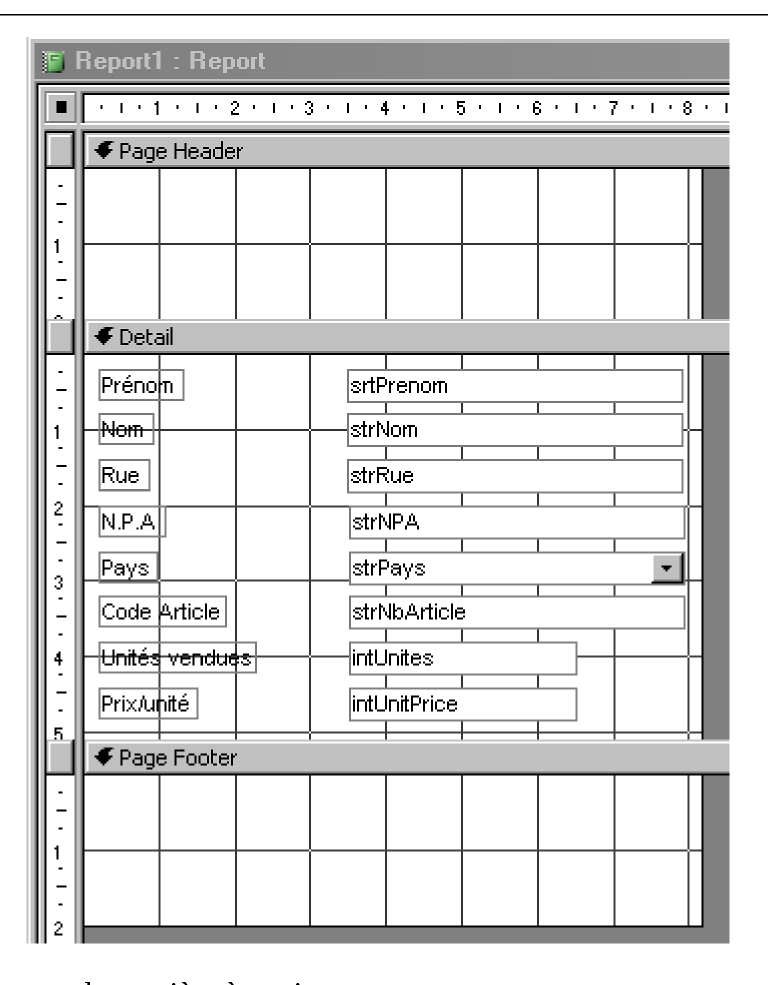
qui donne:


qryFactures : Select Query

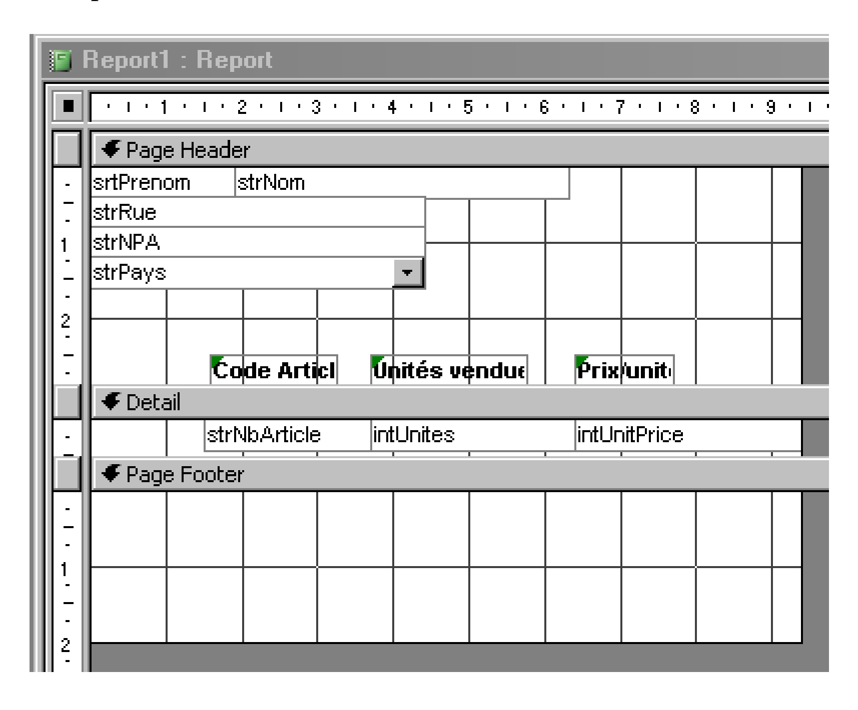
Prénom	Nom	Rue	N.P.A	Pays	Code Article	Unités vendues	Prix/unité
Albert	BOLTZMANN	Ch. des Grottes	1021	Islande	GEN-001	100	SFr. 0.2
Charlie	ANGEL	Arastrasse	3048	Suisse	GEN-003	200	SFr. 25.0
Alain	DUTRON	Ch. de Chandieu	1006	Turquie	INF-002	1500	SFr. 1.3
Alain	DUTRON	Ch. de Chandieu	1006	Turquie	GEN-002	1000	SFr. 0.5
Charlie	ANGEL	Arastrasse	3048	Suisse	GEN-003	50	SFr. 25.0
Albert	BOLTZMANN	Ch. des Grottes	1021	Islande	INF-002	500	SFr. 1.3
Alain	DUTRON	Ch. de Chandieu	1006	Turquie	INF-003	30	SFr. 37.7
Albert	BOLTZMANN	Ch. des Grottes	1021	Islande	GEN-002	50	SFr. 0.5
Charlie	ANGEL	Arastrasse	3048	Suisse	INF-001	5	SFr. 261.4
Charlie	ANGEL	Arastrasse	3048	Suisse	INF-003	10	SFr. 37.7
Charlie	ANGEL	Arastrasse	3048	Suisse	GEN-004	80	SFr. 10.2
Alain	DUTRON	Ch. de Chandieu	1006	Turquie	INF-001	15	SFr. 261.4
Albert	BOLTZMANN	Ch. des Grottes	1021	Islande	INF-003	5	SFr. 37.7
Alain	DUTRON	Ch. de Chandieu	1006	Turquie	GEN-003	30	SFr. 25.0
Charlie	ANGEL	Arastrasse	3048	Suisse	GEN-006	200	SFr. 0.5
Alain	DUTRON	Ch. de Chandieu	1006	Turquie	INF-002	750	SFr. 1.3
Charlie	ANGEL	Arastrasse	3048	Suisse	GEN-001	100	SFr. 0.2
Charlie	ANGEL	Arastrasse	3048	Suisse	GEN-004	30	SFr. 10.2
Albert	BOLTZMANN	Ch. des Grottes	1021	Islande	INF-002	100	SFr. 1.3
Albert	BOLTZMANN	Ch. des Grottes	1021	Islande	GEN-003	30	SFr. 25.0

Record: 1 of 24

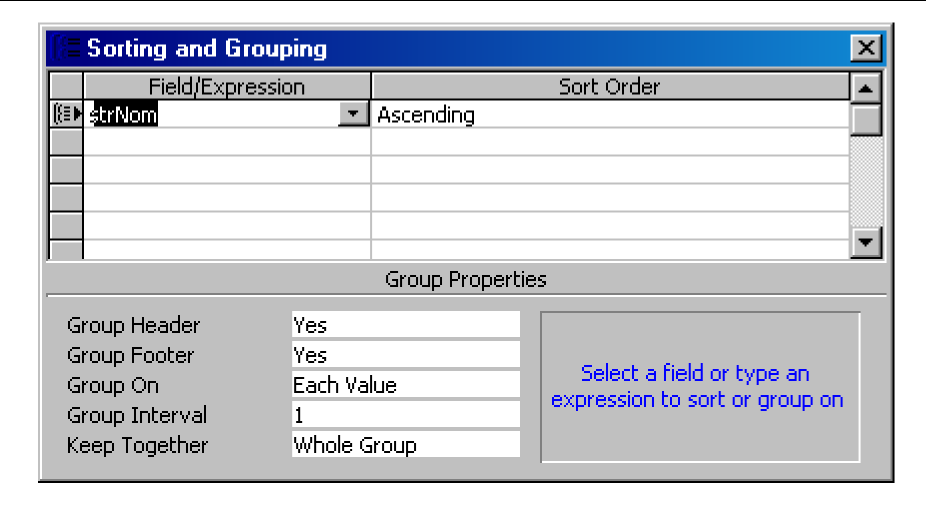
Nous aimerions maintenant construire un état qui affiche par client la liste des produits achetés en ayant un client par page. Pour faire vite, et juste pour voir le principe, créez un rapport automatique de cette requête. Cela vous donnera:



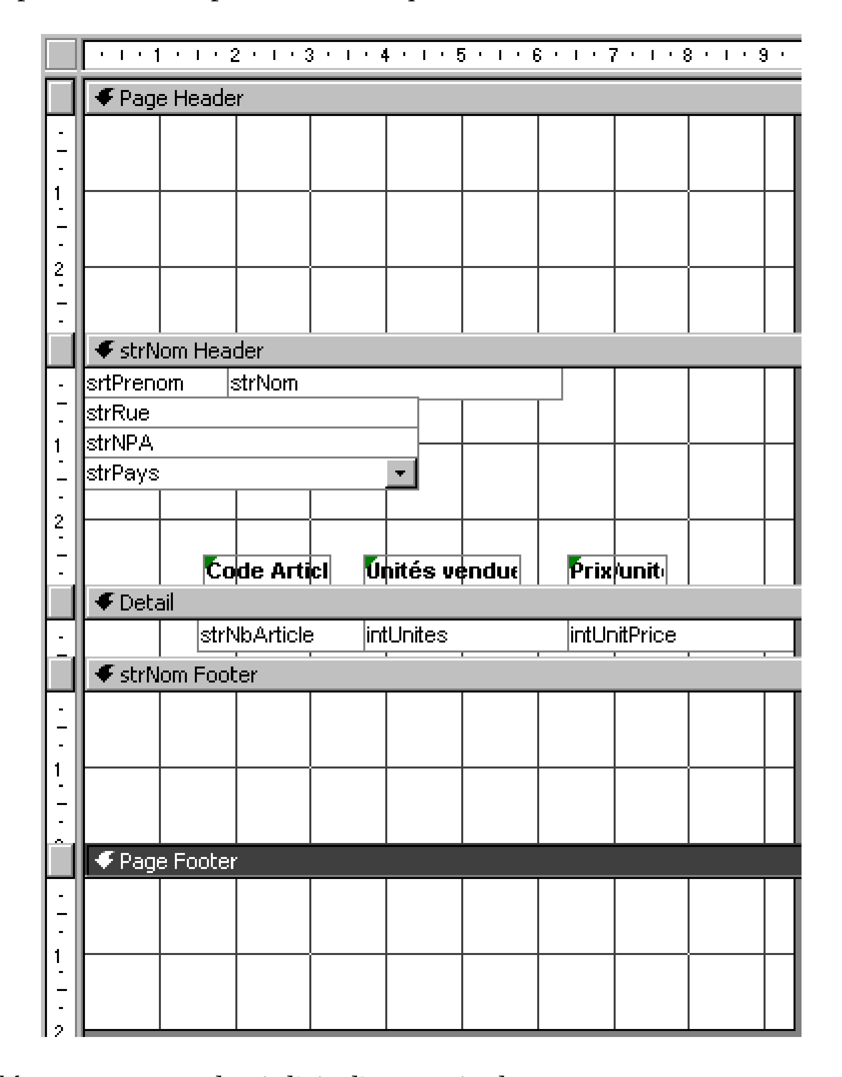
Redisposez les champs de manière à avoir:



Ensuite allez dans le menu *Affichage/Tris et Groupes* et mettez-y les valeurs visibles ci-dessous:



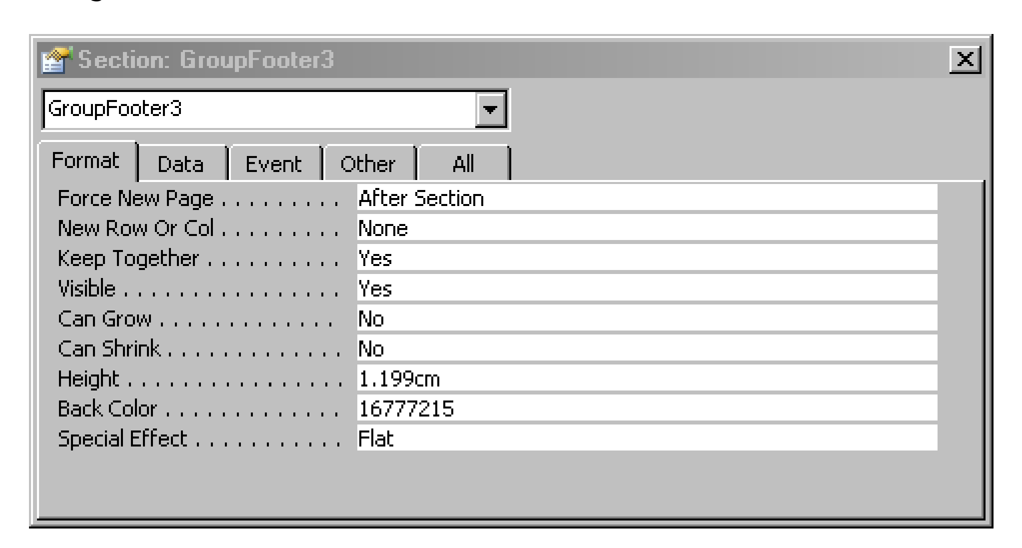
Validez et redisposez les champs comme indiqués ci-dessous:



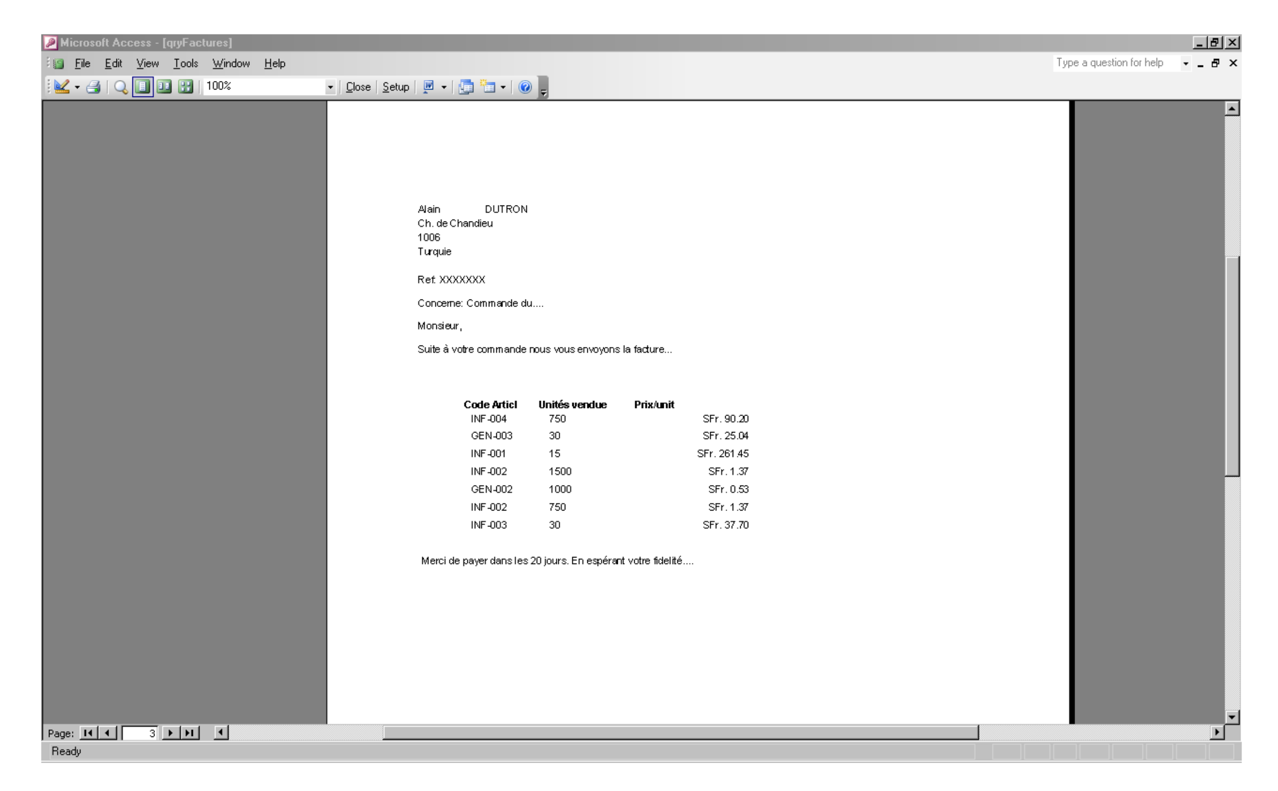
Et enfin complétez pour que cela ait l'air d'une petite lettre:

Page Header
strNom Header
srtPrenom strNom
strRue
strNPA
strPays
Ref: XXXXXXX
Concerne: Commande du....
Monsieur,
Suite à votre commande nous vous envoyons la facture...
Code Articl Unités vendue Prix/unit
Detail
strNbArticle intUnites intUnitPrice
strNom Footer
Merci de payer dans les 20 jours. En espérant votre fidelité....
Page Footer

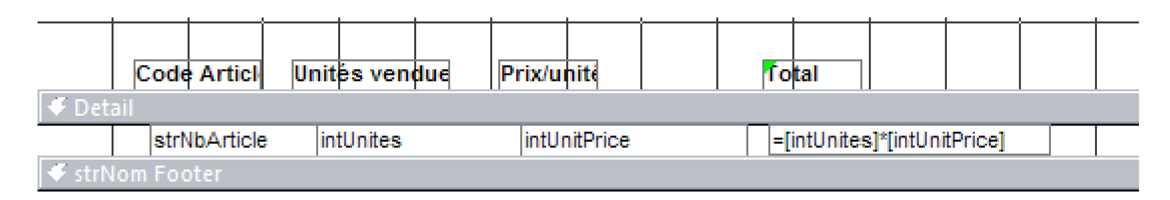
Ensuite, il faut faire un double clic sur la ligne *strNom Footer* et changer la valeur du champ *Force New Page*:



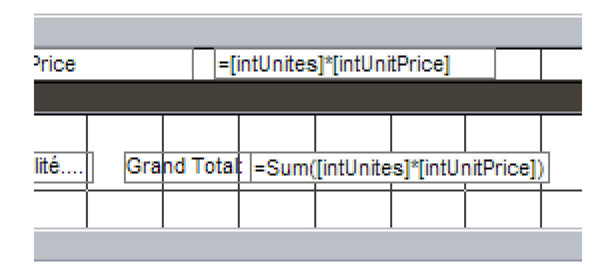
Cela vous donner un état par client qui aura la forme suivant au final:



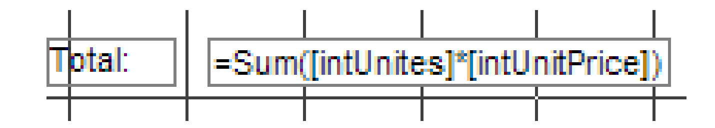
Ensuite, on peut rajouter le total de chaque ligne en ajoutant manuellement un contrôle de type *Field* avec une petite formule:



et enfin le total des totaux:



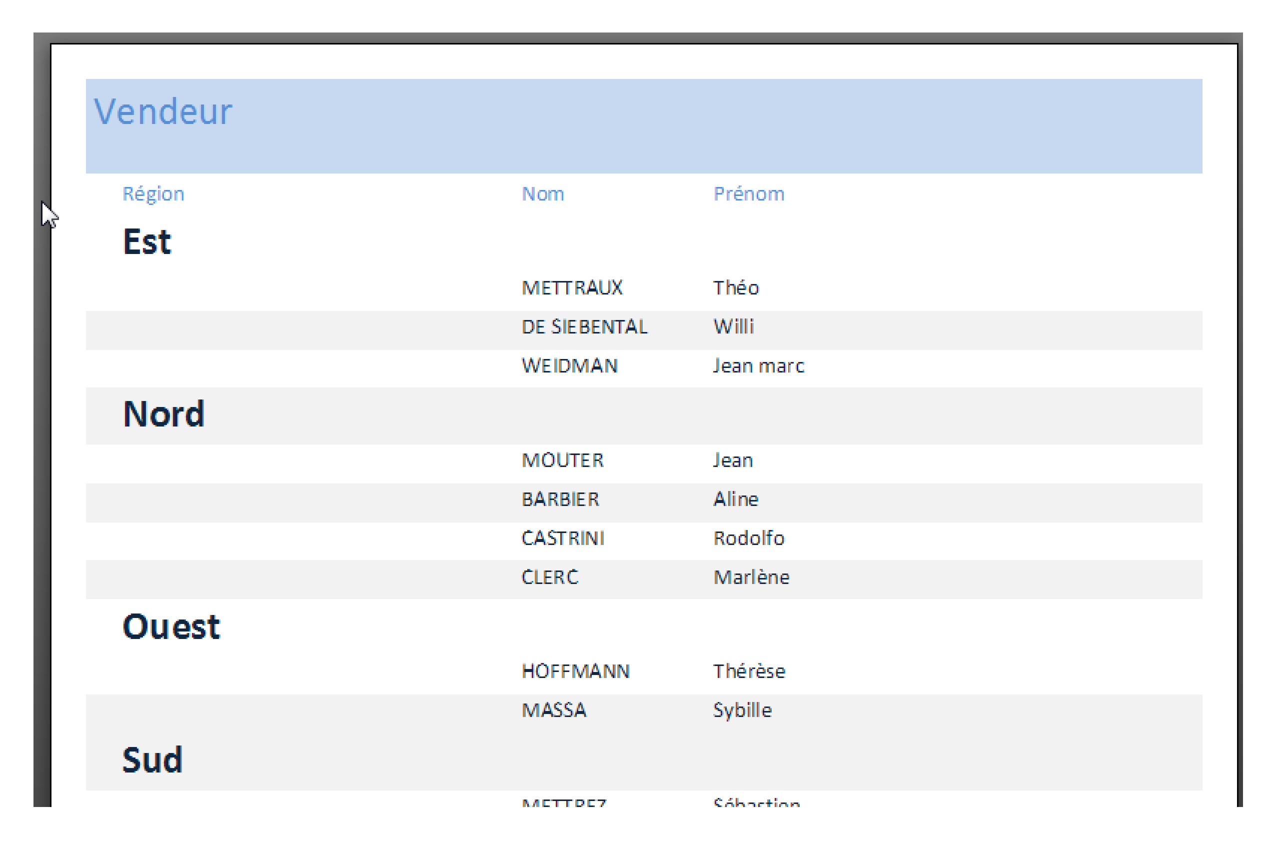
Remarquez bien comment la formule du grand total est écrite:



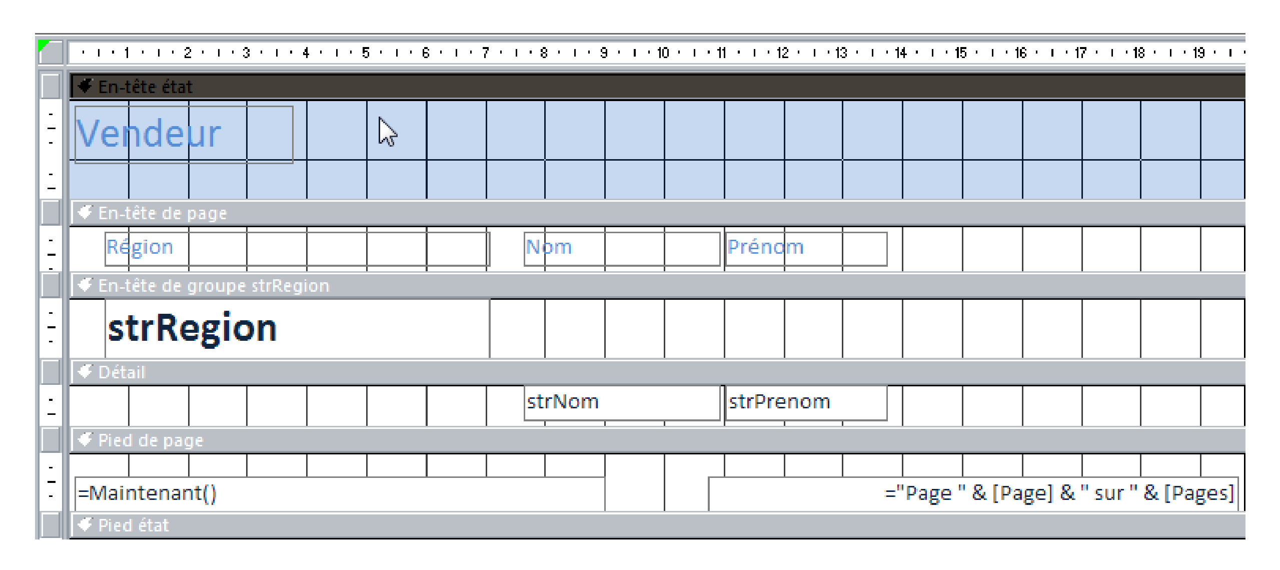
Il n'est pas possible à ce jour dans Access de faire des grands totaux dans des rapports qui utiliseraient le nom d'un champ intermédiaire. Donc ceci par exemple, ne fonctionnera pas:

Nous allons voir ici un des pièges classiques et merveilleux de MS Access....

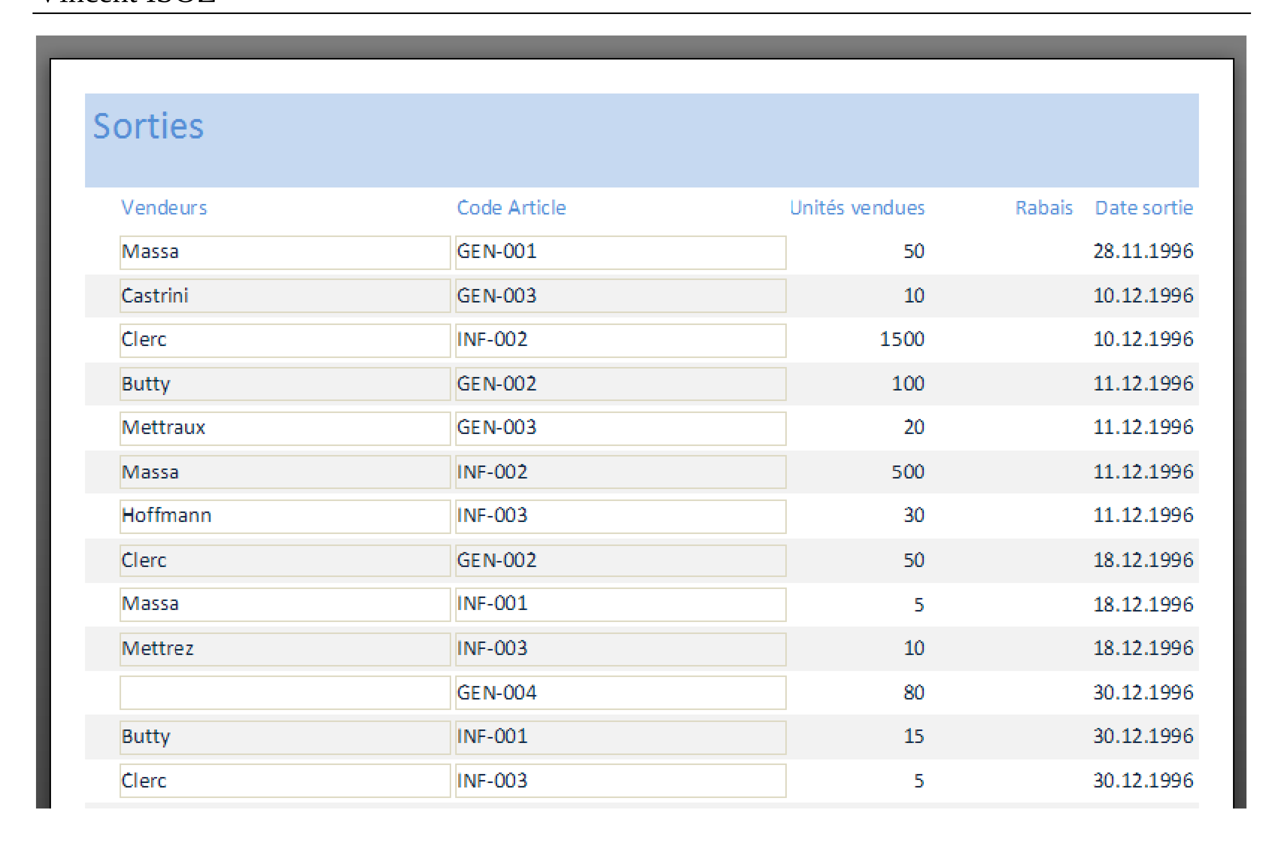
Créez d'abord l'état suivant de façon classique avec l'assitant d'État sur la base de la table *tblVendeurs*:



Soit au niveau structurel:



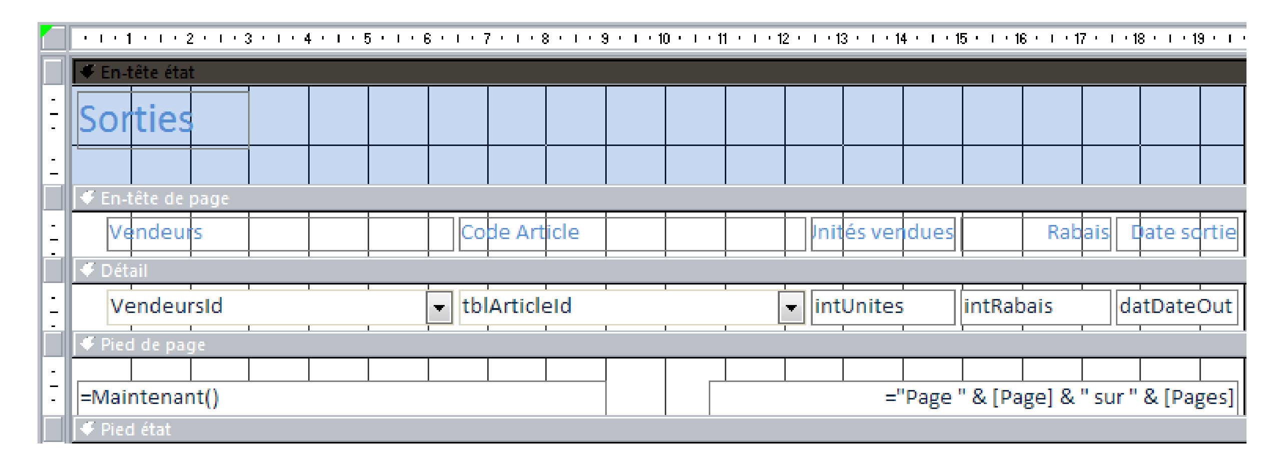
Maintenant, créez l'état suivant qui affiche les sorties et nommez-le *srptSortiesParVendeur*:



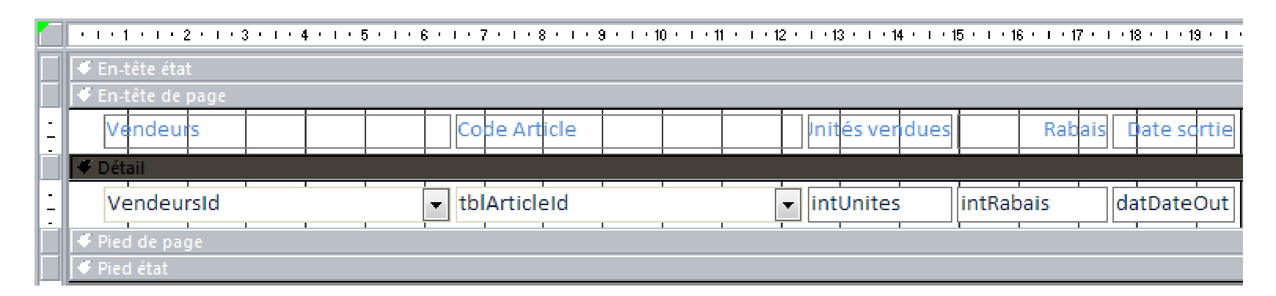
Sorties

Vendeurs	Code Article	Unités vendues	Rabais	Date sortie
Massa	GEN-001	50		28.11.1996
Castrini	GEN-003	10		10.12.1996
Clerc	INF-002	1500		10.12.1996
Butty	GEN-002	100		11.12.1996
Mettraux	GEN-003	20		11.12.1996
Massa	INF-002	500		11.12.1996
Hoffmann	INF-003	30		11.12.1996
Clerc	GEN-002	50		18.12.1996
Massa	INF-001	5		18.12.1996
Mettrez	INF-003	10		18.12.1996
	GEN-004	80		30.12.1996
Butty	INF-001	15		30.12.1996
Clerc	INF-003	5		30.12.1996

Soit au niveau structurel:

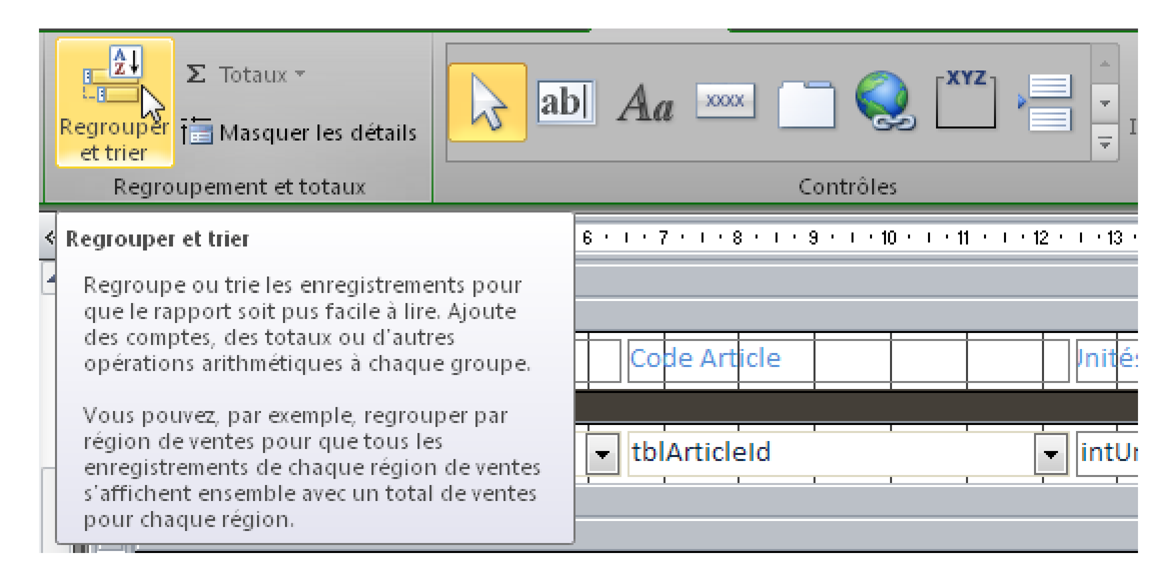


Nous souhaitons maintenant mettre ce dernier rapport en tant que sous-rapport du rapport précédent. C'est-à-dire que pour chaque vendeur, nous puissions voir la liste des éléments vendus. Pour cela le début est simple et classique, nous préparons ce dernier rapport en lui enlevant l'en-tête d'état et le pied de page tel que:

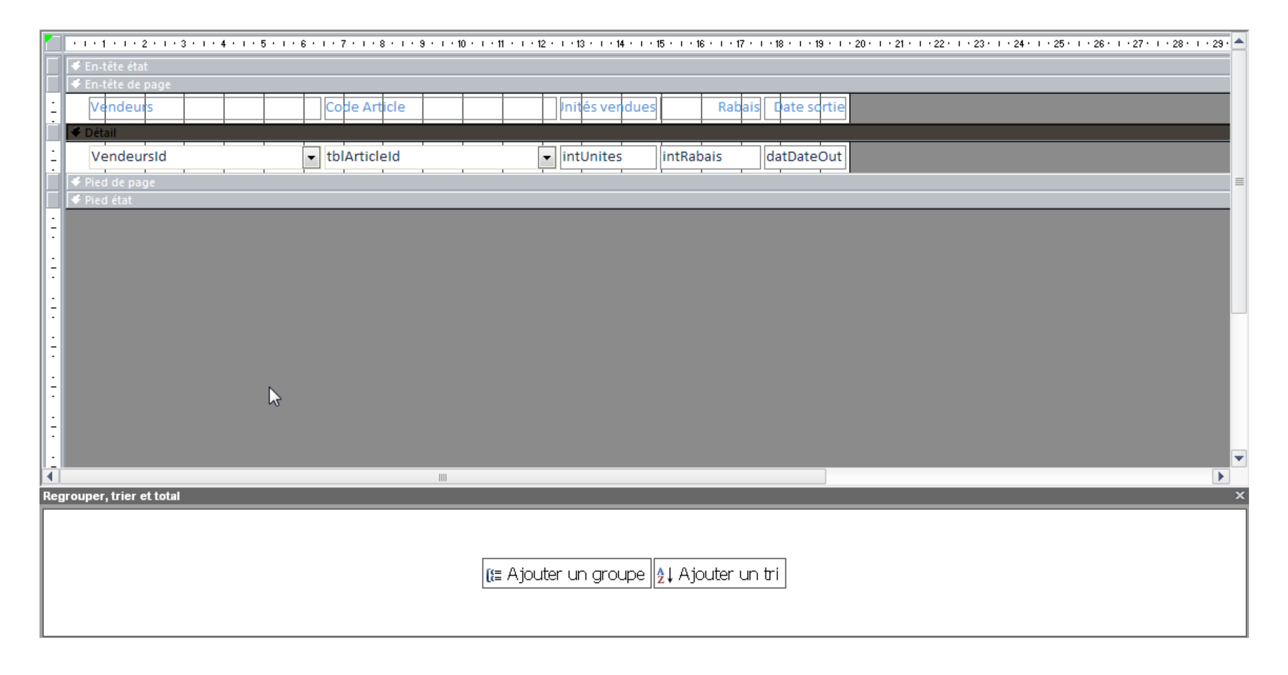


Ensuite **vient maintenant le piège**!!! Effectivement, si vous insérez ce rapport en tant que sous-rapport, l'en-tête de page avec les intitulés de colonnes **n'apparaîtra pas** (ce relativement logique car... ce n'est justement pas un page).

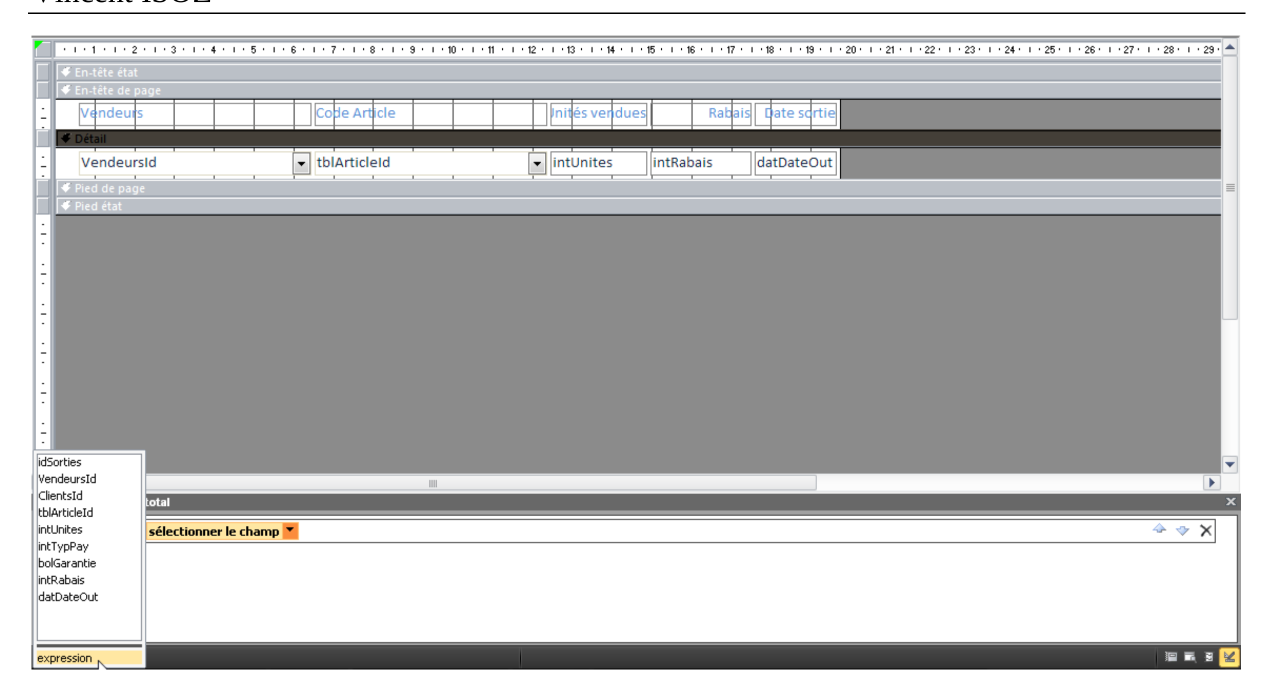
Pour y remédier, il faut créer un groupement. Donc on clique sur le bouton *Regrouper et trier*:



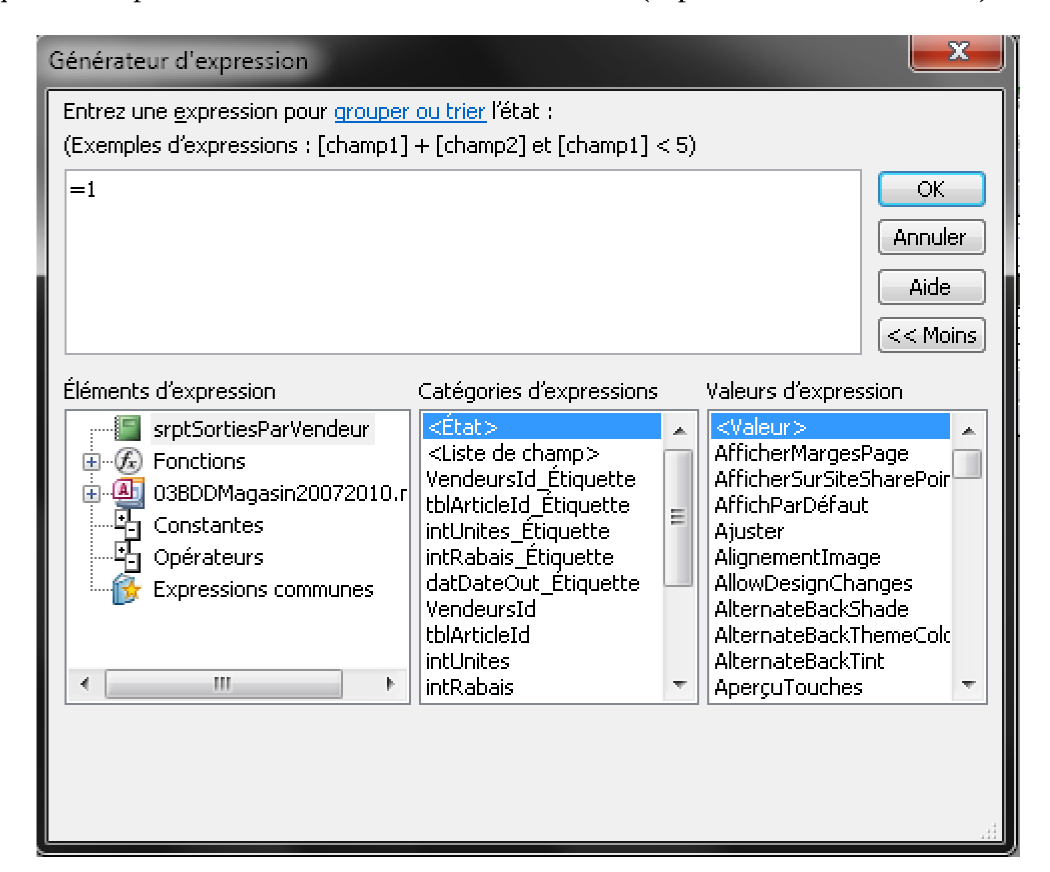
Il vient alors:



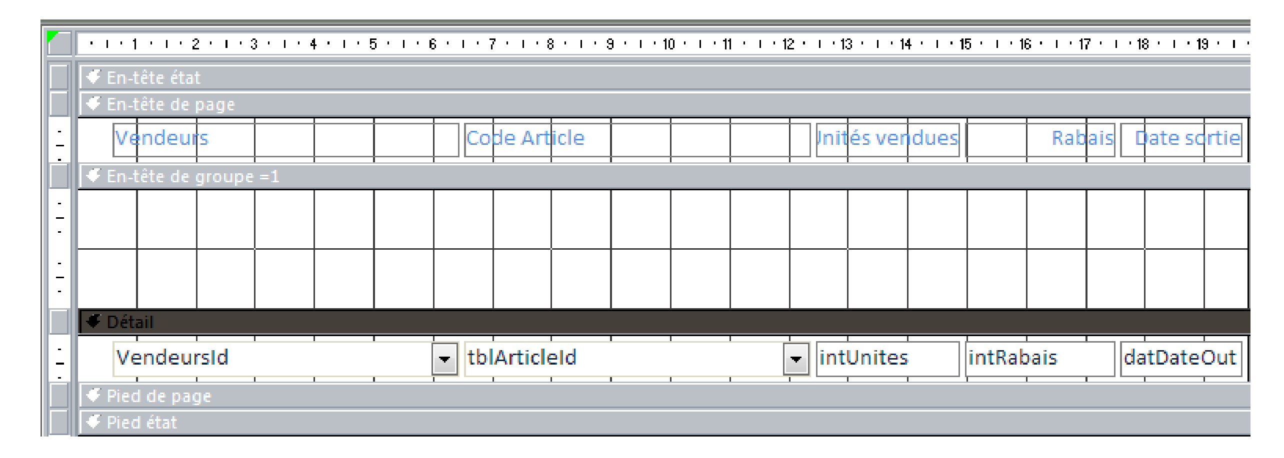
Cliquez sur *Ajouter un groupe*:



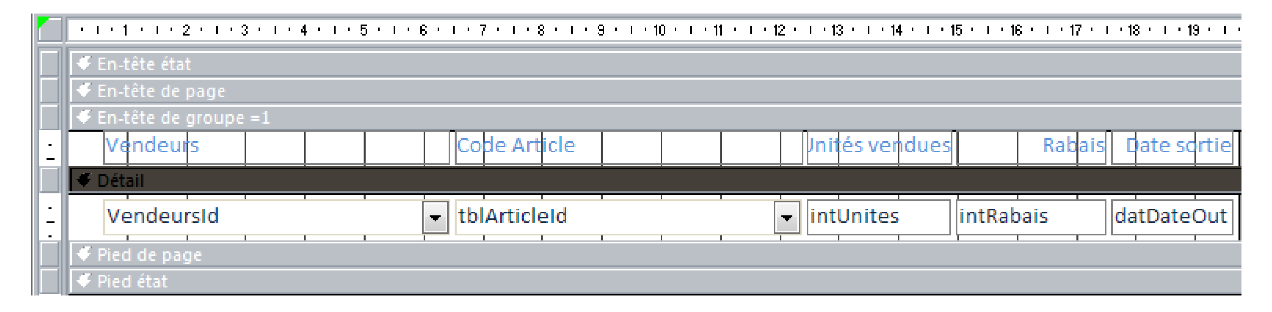
Et cliquez sur *expression* et écrivez la formule suivante (super... il fallait deviner...):



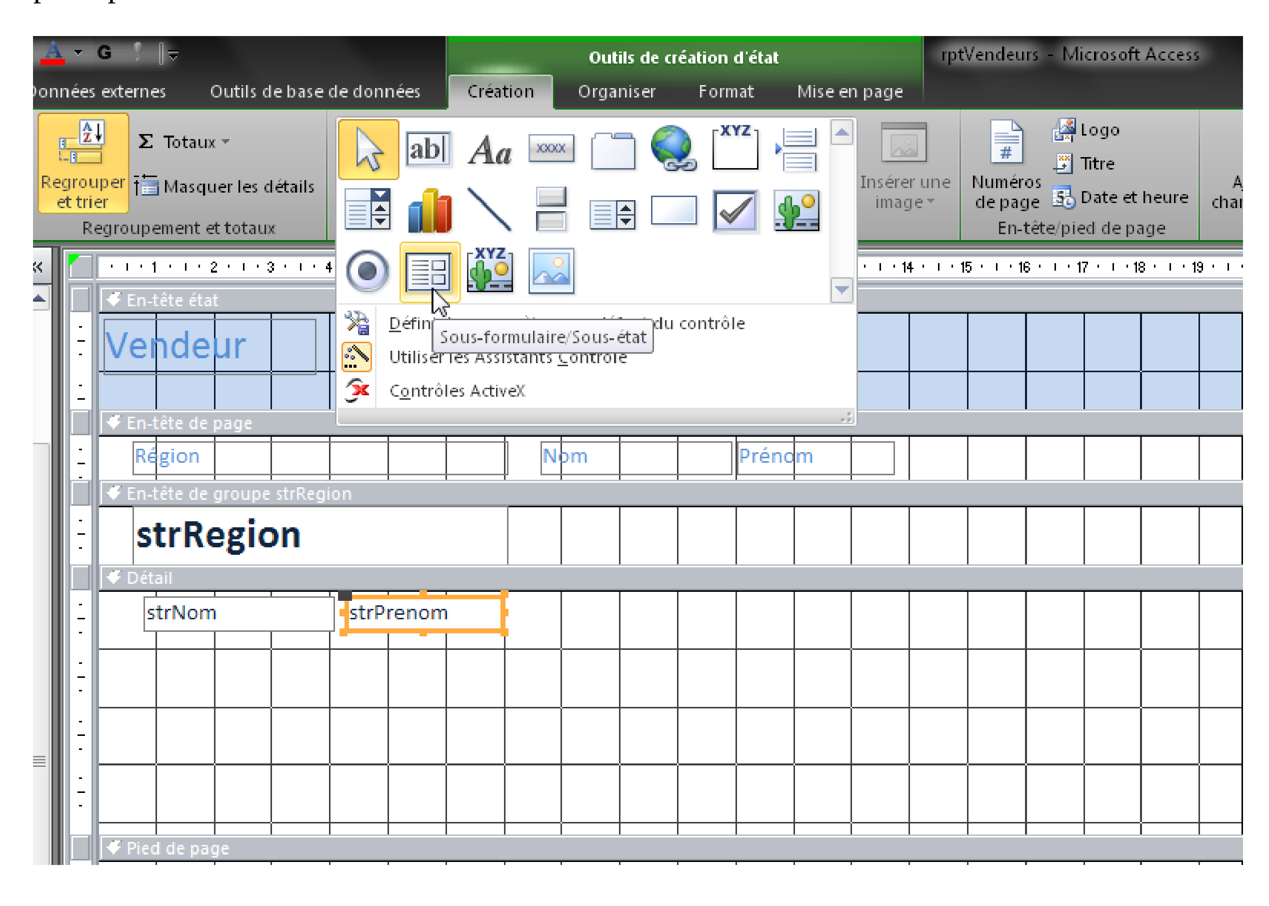
Validez par *OK* et vous aurez:



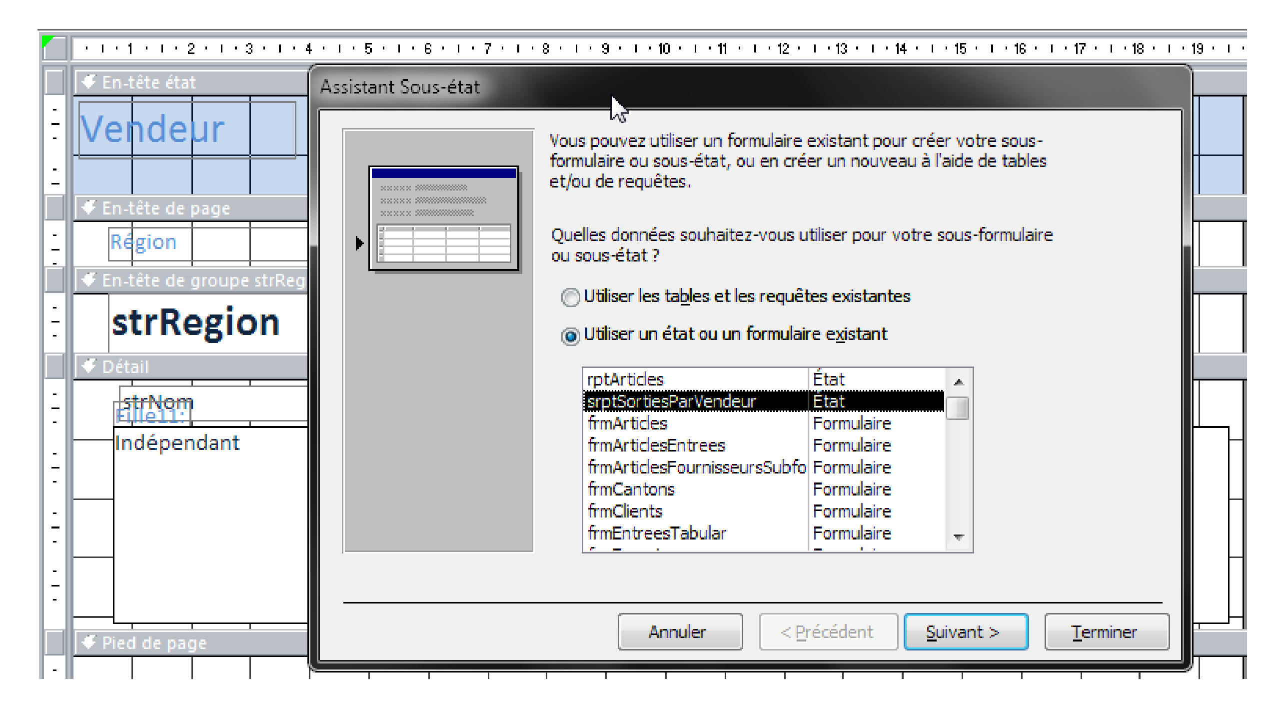
Déplacez les libellés dans la zone d'en-tête à la zone de groupe et réduisez ensuite la zone d'en-tête car elle vient inutile:



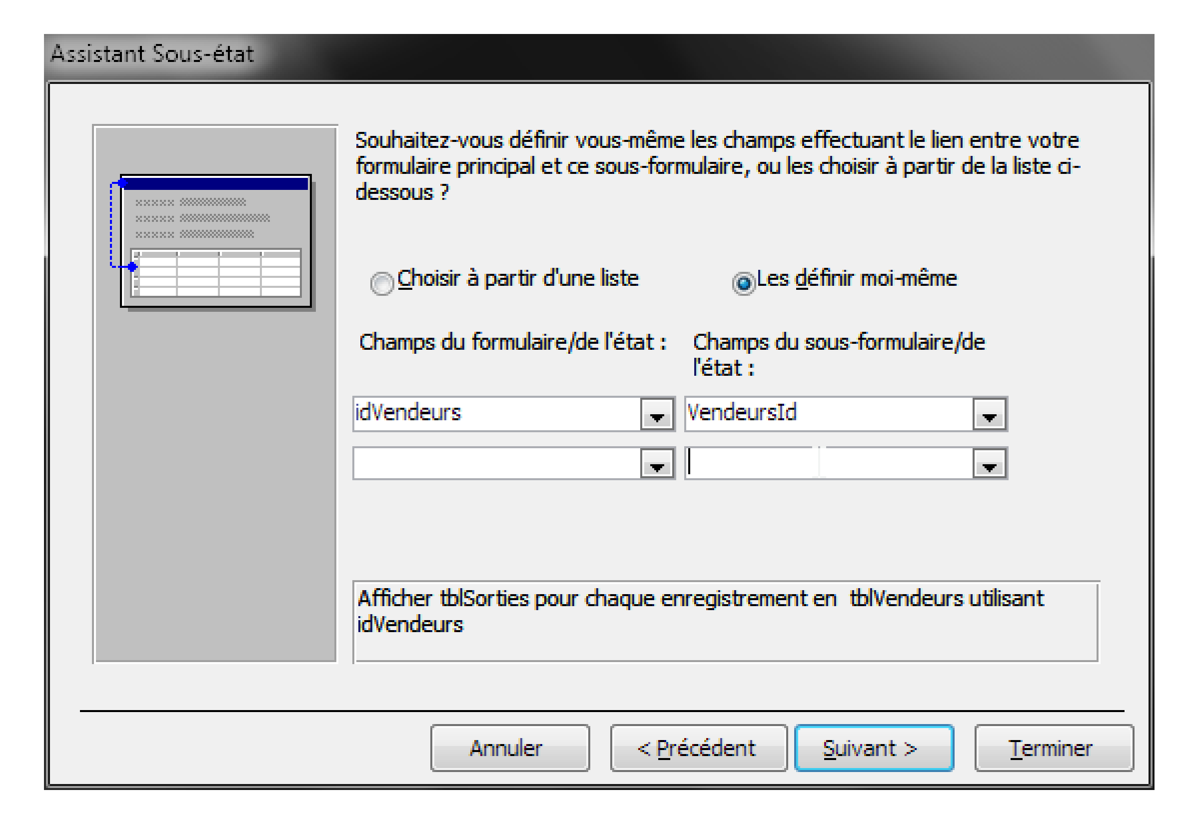
Ensuite, vous pouvez voir que cela fonctionner. Ajouter ce sous-rapport dans le rapport principal:



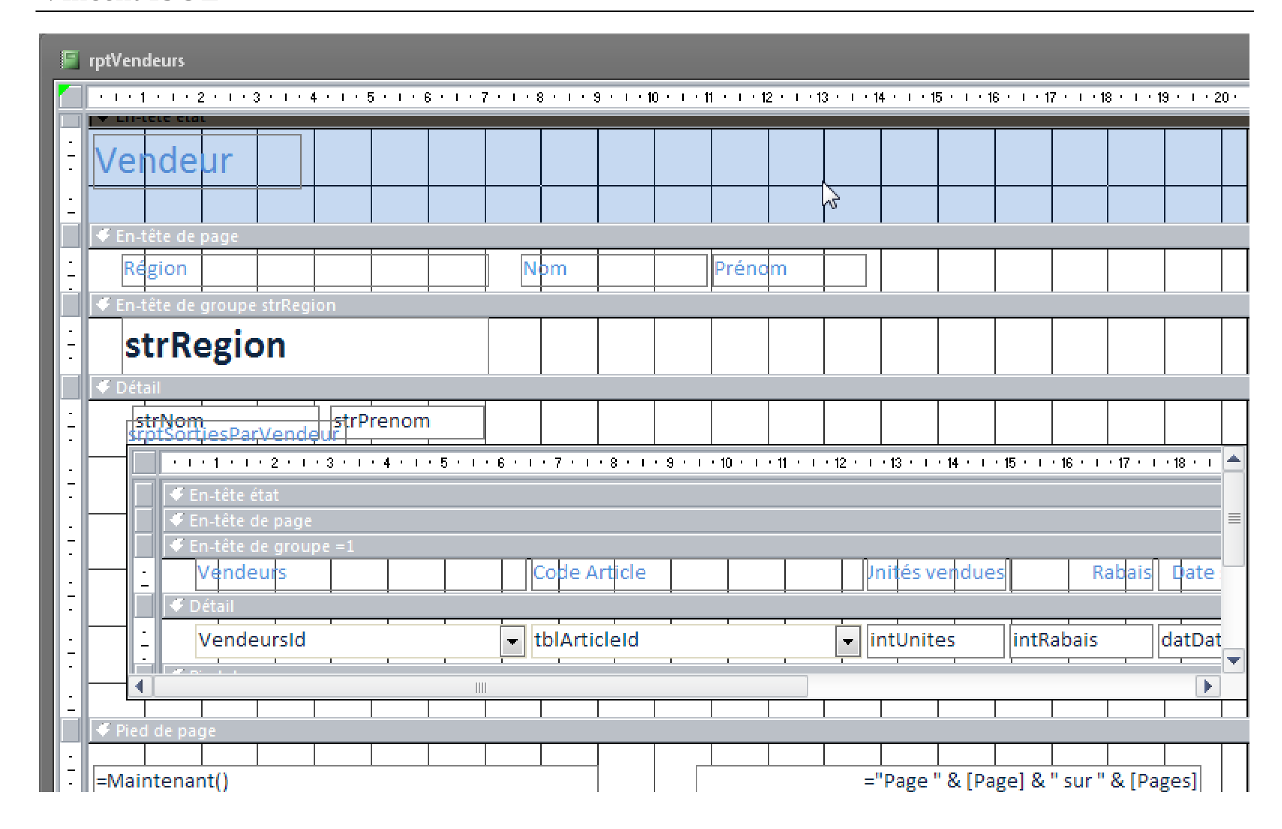
Ensuite nous dessinons la zone du sous-rapport l'assistant démarre:



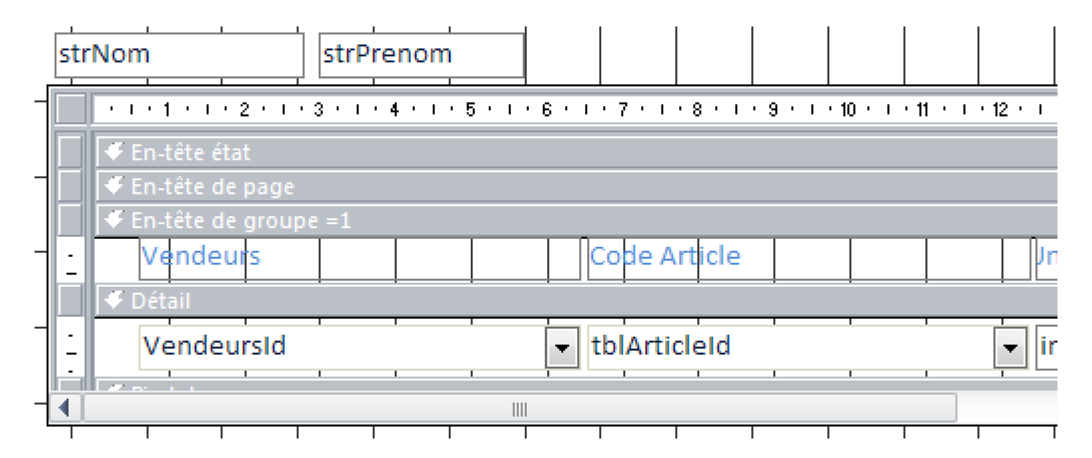
Nous sélectionnons *srptSortiesParVendeur* et cliquons sur *Suivant* pour définir les clés qui feront la liaison:



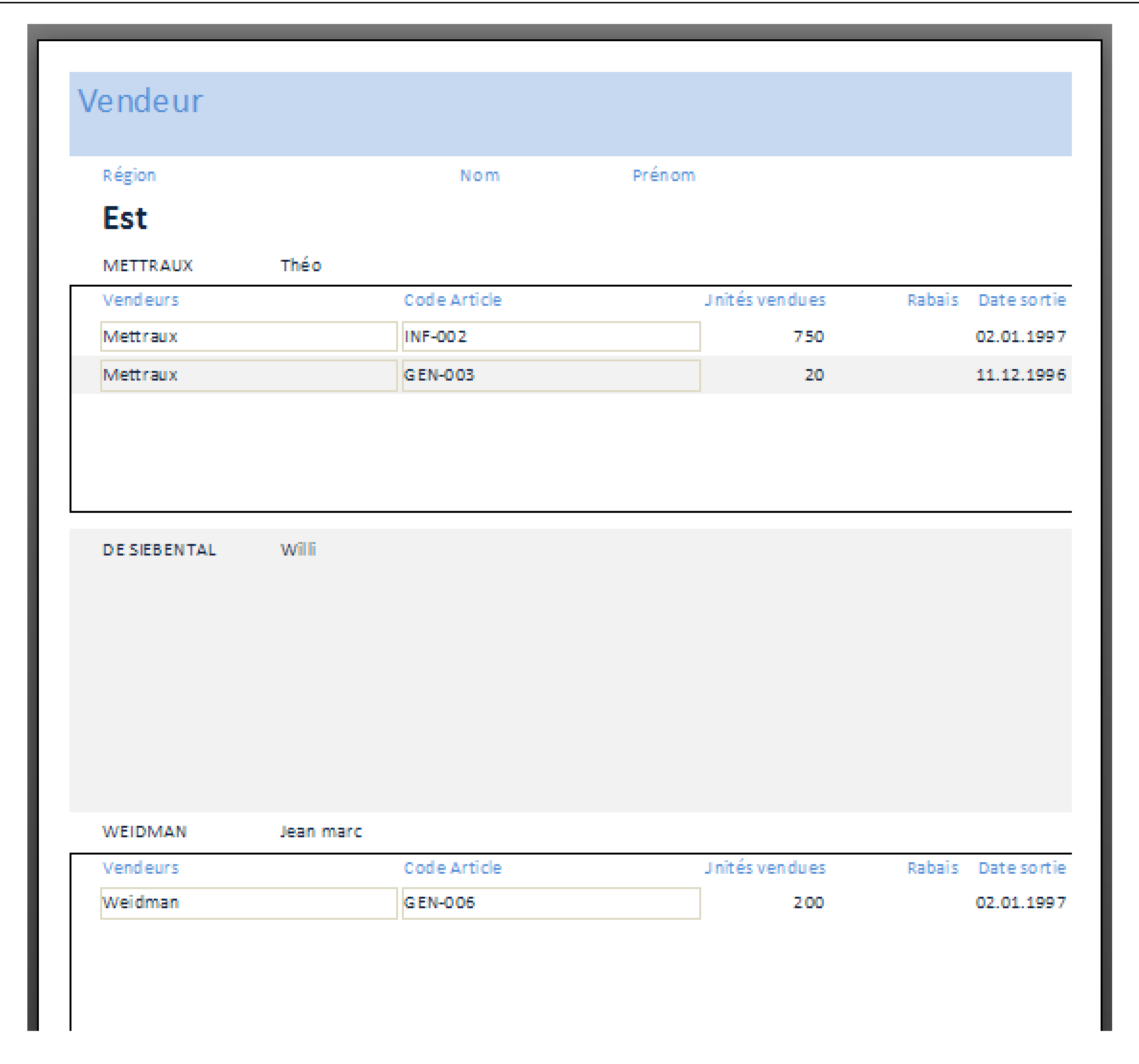
Cliquez sur *Terminer*, ce qui donnera:



Supprimez la petite légende bleue dans le coin supérieur droit où est écrit le nom du sous-rapport:

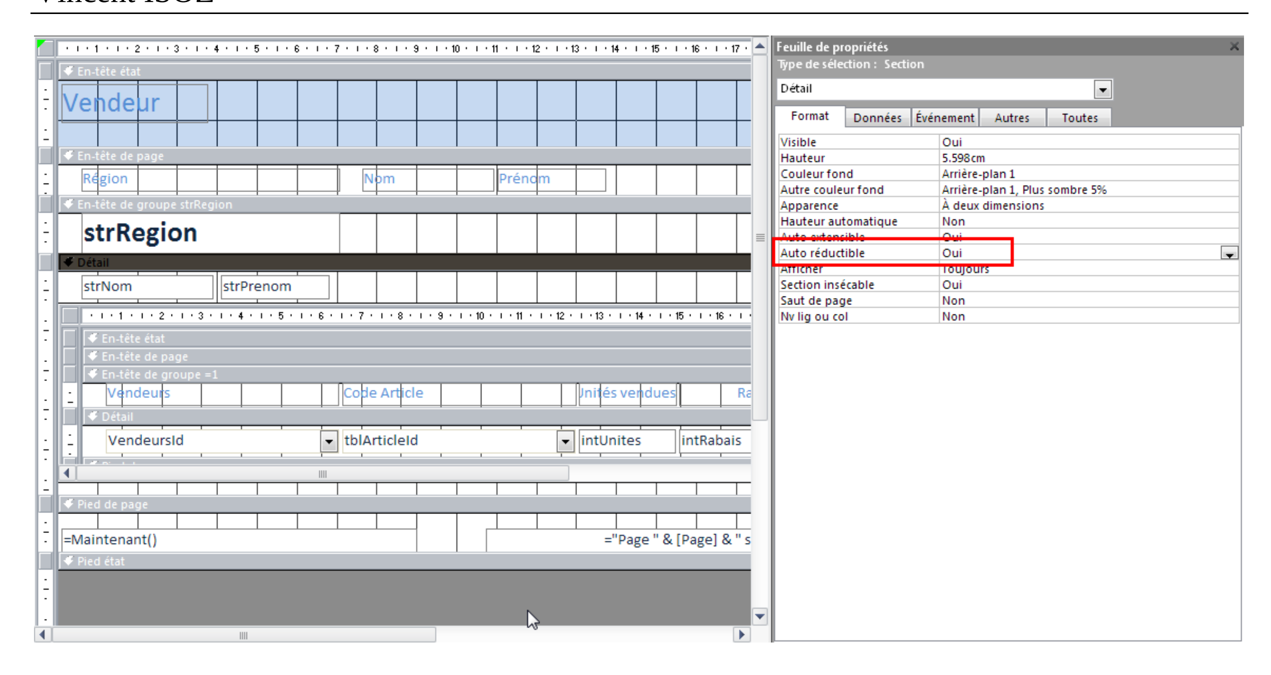


et lancez l'aperçu:



Donc nous retrouvons bien les intitulés es colonnes dans le sous-rapport. Donc le problème est réglé mais il en apparaît un autre... Les vendeurs qui n'ont aucune vente occupent une zone verticale vide innaceptable!

Pour corriger cela, nous éditons le rapport principal et dans les propriétés de la section *Détails*:



Nous mettons la propriété *Auto réductible* à la valeur *Oui*. Ce qui donnera au final:

Vendeur

Région Nom Prénom

Est

METTRAUX Théo

Vendeurs	Code Article	Unités vendues	Rabais	Date sortie
Mettraux	INF-002	750		02.01.1997
Mettraux	GEN-003	20		11.12.1996

DE SIEBENTAL Willi

WEIDMAN Jean marc

Vendeurs	Code Article	Unités vendues	Rabais	Date sortie
Weidman	GEN-006	200		02.01.1997

Nord

MOUTER Jean

BARBIER Aline

CASTRINI Rodolfo

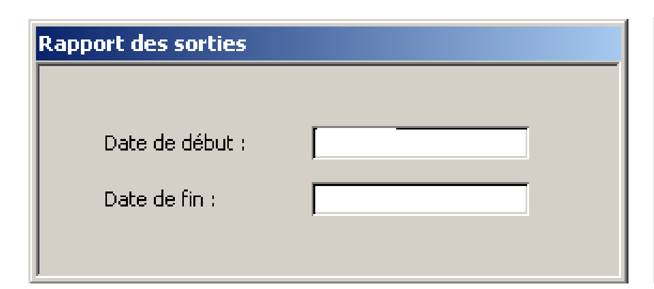
Vendeurs	Code Article	Unités vendues	Rabais	Date sortie
Castrini	GEN-004	30		08.01.1997
Castrini	GEN-003	30		02.01.1997
Castrini	GEN-003	10		10.12.1996

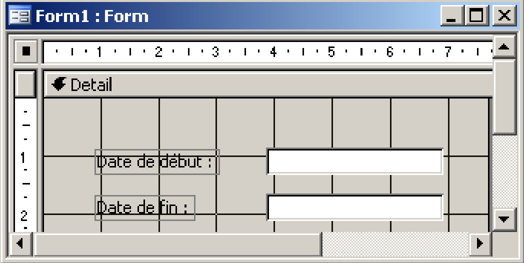
CLERC Marlène

Vendeurs	Code Article	Unités vendues	Rabais	Date sortie

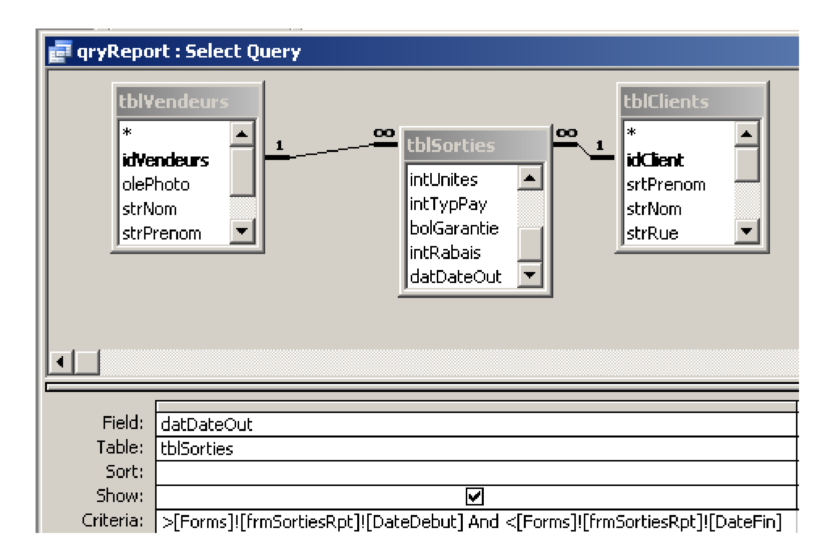
8.5 Rapport paramétré par formulaire

Créez maintenant un formulaire identique à celui ci-dessous que vous nommerez *frmSortiesRpt* (veuillez nommer respectivement les deux champs *fldDateDebut* et *fldDateFin*):

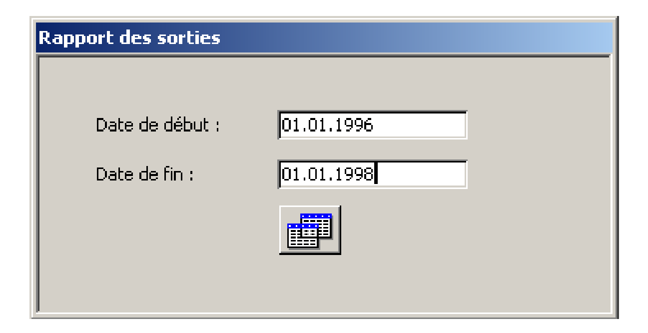




Dans la requête, effectuez les modifications suivantes:

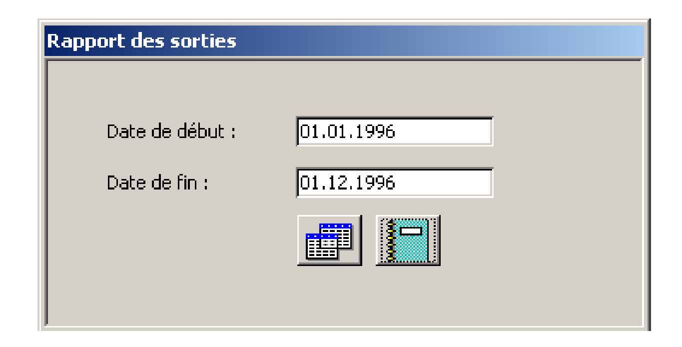


et ajoutez sur le formulaire, un bouton qui permet d'exécuter la requête (quand vous avez terminé de saisir la dernière date, faites de préférence un TAB pour valider la date):



Attention! Une curiosité de l'ouverture d'états en passant par les formulaires c'est que suivant le type de driver d'imprimante que vous aurez d'installé sur votre PC vous aurez systématiquement un message d'erreur du type *OpenReport Canceled*.

Voilà une interface bien plus conviviale n'est-ce pas ? Maintenant enlevez le bouton de la requête et ajoutez-y celui du rapport (pour vérifier que lui aussi se base sur le contenu de ce formulaire):



Une fois que vous avez testé la fonctionnalité de la précédente requête. Ajoutez maintenant (seul !) une liste déroulante dans le formulaire créé précédemment permettant de choisir le vendeur pour lequel vous souhaitez avoir un rapport (temps estimé: 5 minutes)

8.6 Objet ActiveX

Même si c'est mieux qu'avant (un seul formulaire pour les deux questions, pas besoin de relancer le tout depuis le début) il y a toujours les dates qu'il faut saisir (ce qui est fort dommageable…). Remédions à cela et ajoutons un bouton à côté de chaque champ qui permette de sélectionner la date dans un calendrier:

Procédure:

1. Créer deux nouveaux formulaires vierges que vous enregistrerez sous les noms respectifs *frmCalendrierDateDebut* et *frmCalendrierDateFin*.

2. Passer le formulaire en mode création

3. Cliquer sur le bouton

4. Insérer un objet du type: Contrôle Calendrier X.0 dans chacun des formulaires (nommez le calendrier: bouton droit *Propriétés*):

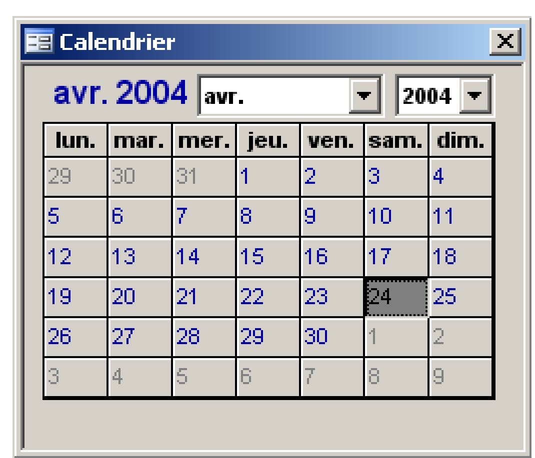
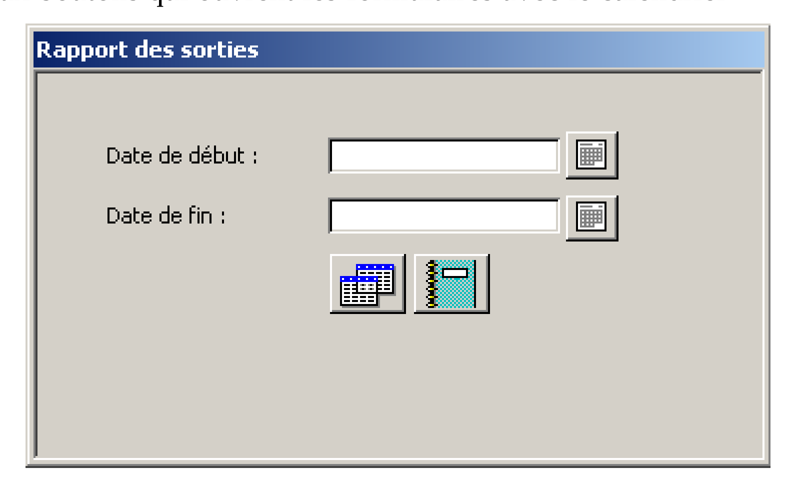


Figure 9 Calendrier Access

1. Fermer et enregistrer le formulaire (on vous laisse le choix du nom du formulaire contenant le calendrier)

2. Aller dans le formulaire où l'utilisateur doit "piquer" les dates

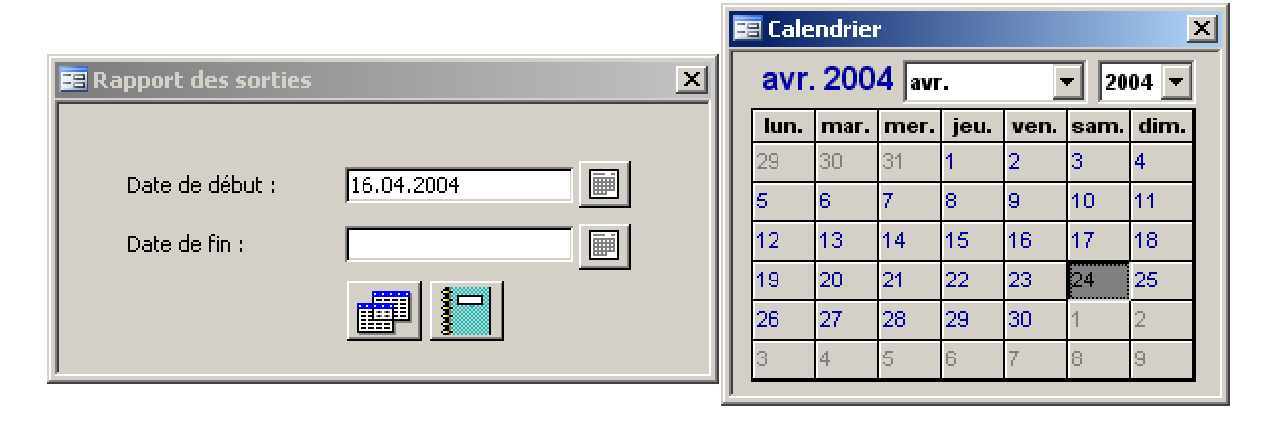
3. Créer deux boutons qui ouvrent les formulaires avec le calendrier



Dans chacun des formulaires contenant le calendrier insérer le code suivant dans l'événement *OnClose:*

```
Private Sub Form_Close()
   Forms!frmNomFormulaireAvecDate!NomChamp = NomCalendrier.Value
End Sub
```

et voilà:



Mais que se passe-t-il s'il n'y aucune donnée entre ses deux date ? Eh bien quelque chose de très moche et pas professionnel du tout avec des #Erreur un peu partout ! Comment remédier à cela esthétiquement. Et bien, dans les propriétés du rapport il existe un événement appelé *On No Data* , et associez-y le code VBA suivant:

```
Private Sub Report_NoData(Cancel As Integer)
   MsgBox "Il n'y pas de données. Désolé !"
   DoCmd.CancelEvent
End Sub
```

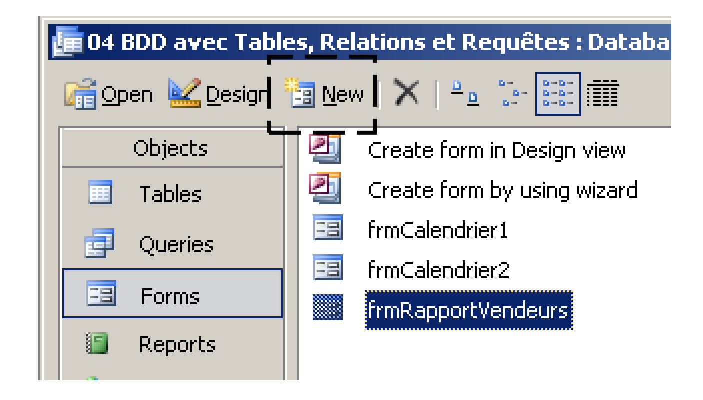
8.7 Graphique statistique inséré

Le but de cet exercice est de créer un graphique qui s'adapte automatiquement sur la base de la requête *qrySommeSorties* et d'inclure ce formulaire:

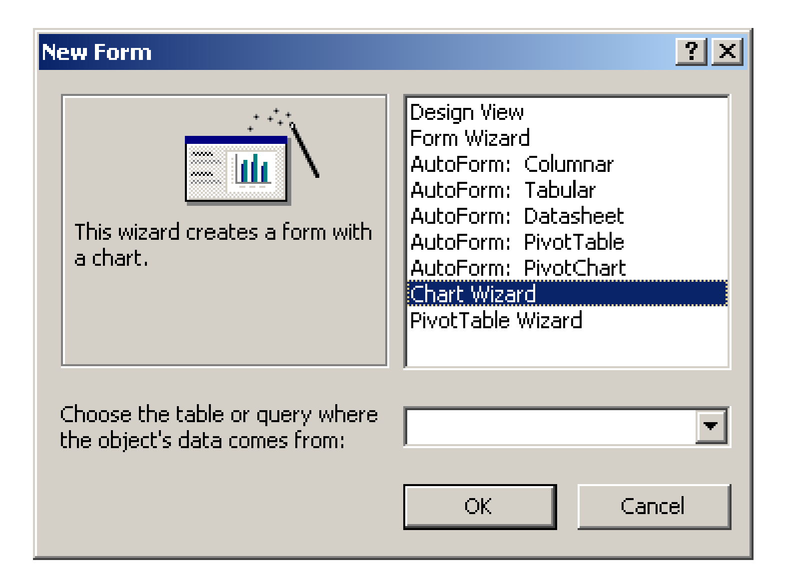
1. Dans un autre formulaire (si ! si !)

2. Dans un état (aussi ….)

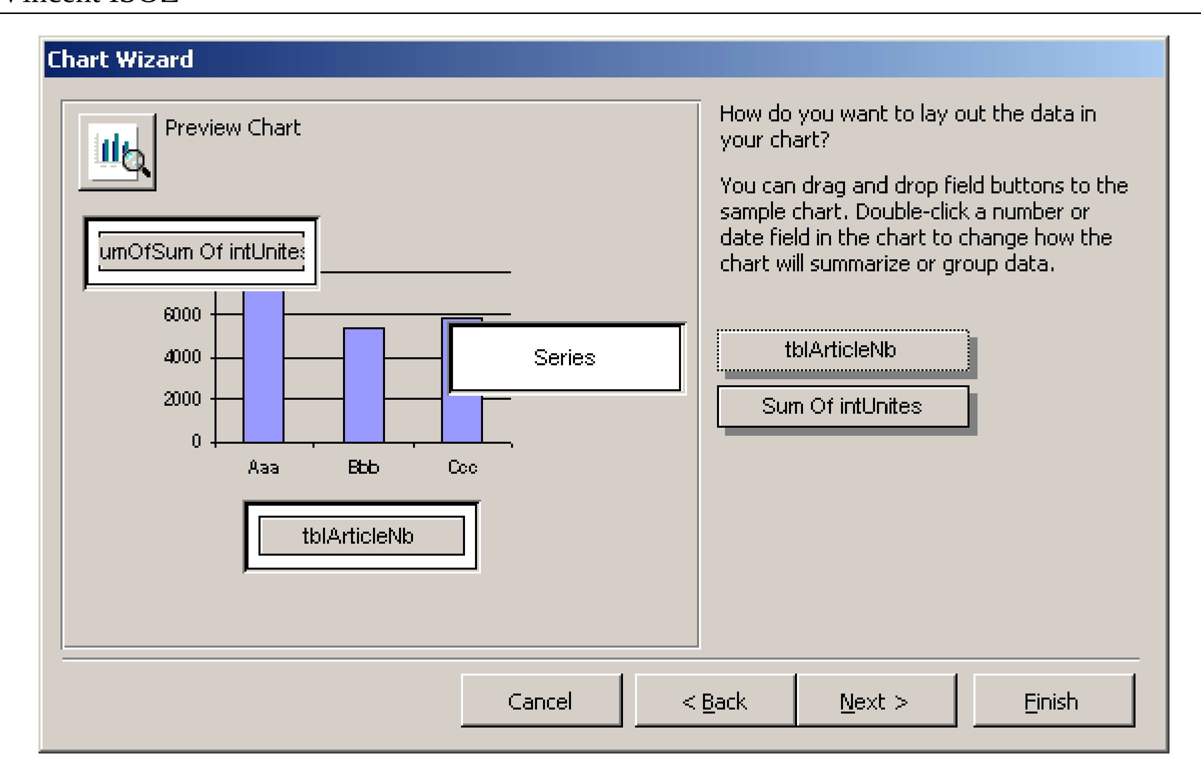
Pour créer un formulaire contenant un graphique ou un tableau croisé dynamique, il faut passer par le bouton mis en évidence ci-dessous (cela n'est pas proposé dans les boutons du dessous curieusement):



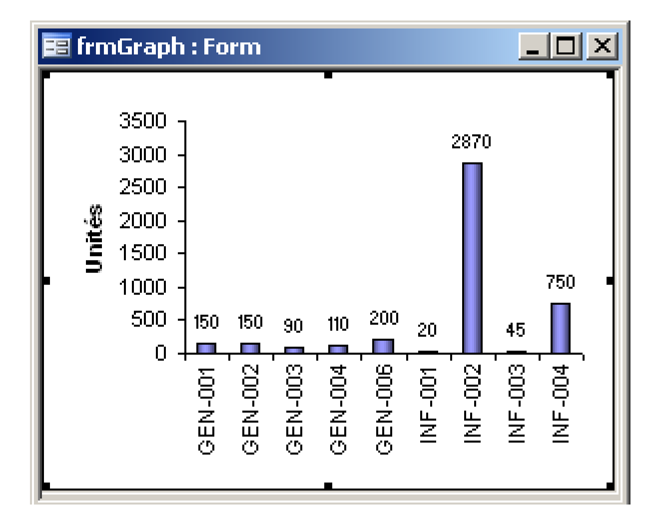
Après quoi, choisissez *Assistant Graphique* (vous remarquerez que certaines des options ci-dessous ne sont disponibles que sous MS Access 2002 ou 2003):



Dans l'assistant, prenez la requête *qrySommeSorties* et choisissez les champs *tblNbArticles* et *Sum of intUnites*:

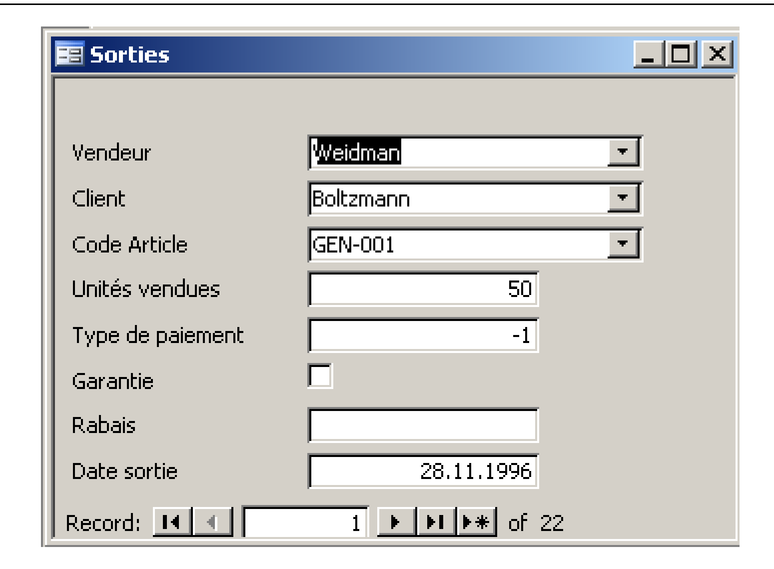


Il faut faire en sorte que le graphique ne soit pas trop gros (5x5 cm au maximum pour notre exercice). Après un peu de travail (le graphique est enregistré sous le nom *frmGraph*):

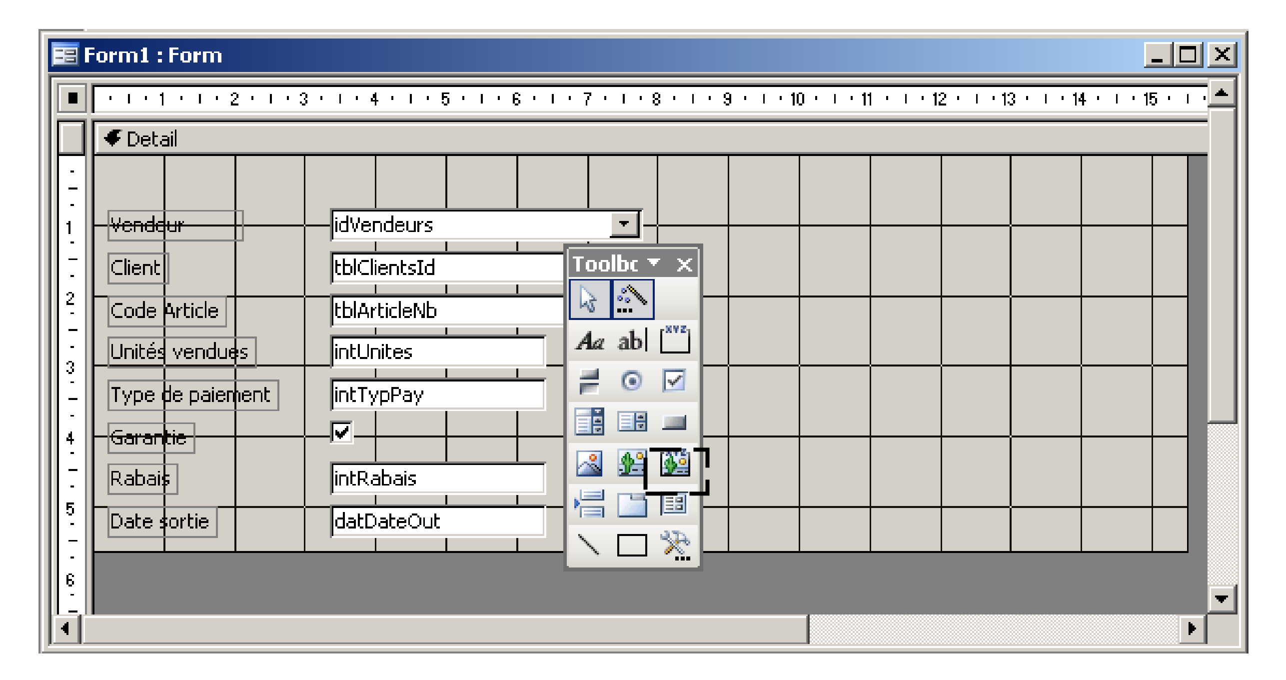


Voilà comment l'on crée un simple graphique dans MS Access… Evidemment, on peut laisser fonctionner l'imaginaire pour personnaliser le formulaire dans lequel se trouve notre graphique (nous avons déjà vu comment faire cela auparavant) et le graphique lui-même.

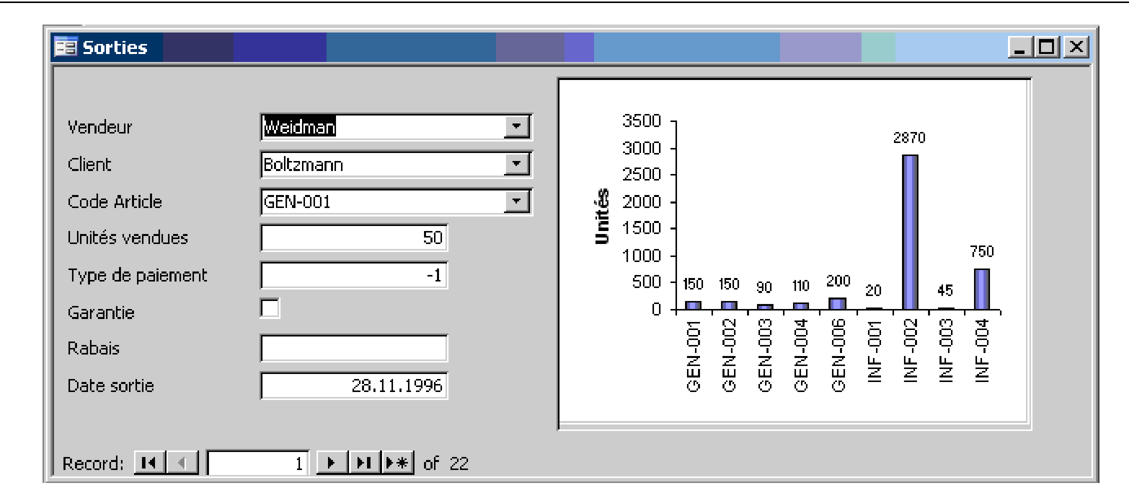
Sinon, le reste tient plutôt à un cours MS Excel qu'à un cours MS Access bien évidemment. Nous allons maintenant voir comment ajouter ce graphique dans un formulaire ou état existant. Créez un formulaire (pas trop moche) de la table *tblSorties*:



Allongez en mode création un tant soit peu en longueur ce formulaire et cliquez sur le bouton suivant de la barre d'outils:

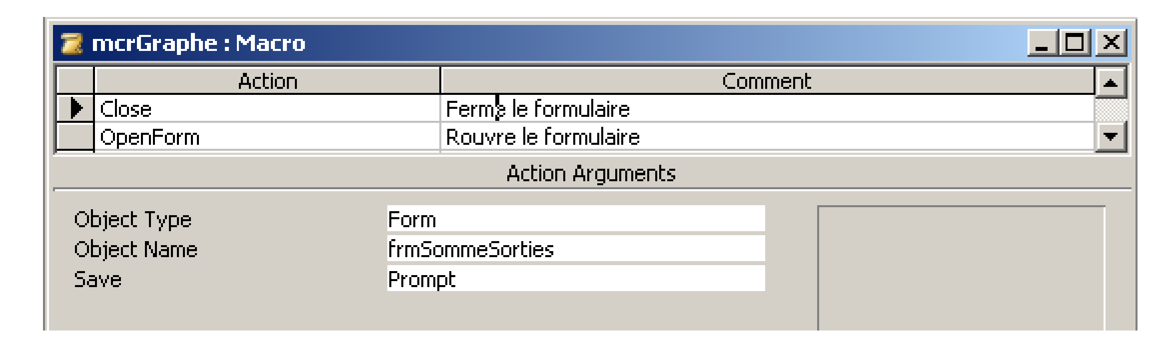


Suivez l'assistant après avoir cliqué sur le bouton *Insérer un sous-formulaire* et choisissez le formulaire contenant le graphique:

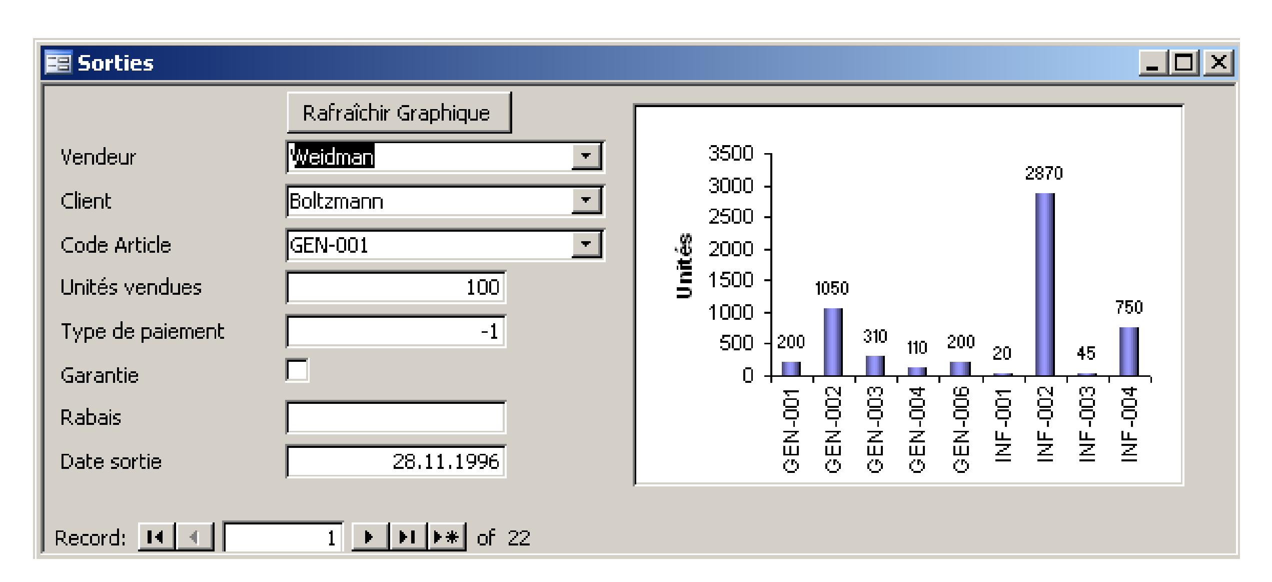


Vous voyez comme cela est fort intéressant. De plus à chaque fois que vous passez d'un enregistrement à l'autre, le graphique reste à l'écran. Par contre, si vous ajoutez des sorties, ou modifiez la valeur des unités vendues, le graphique ne change pas. Pour remédier à cela, vous pouvez créer un bouton rattaché à une macro qui rafraîchira le formulaire en le fermant et en le rouvrant automatiquement.

La macro:



Le formulaire avec le bouton rattaché à la macro:



8.8 Graphique inséré

Créez une copie du formulaire de l'exercice précédent et effacez-y le graphique.

Objectif: créer un graphique dans le formulaire qui pour chaque vendeur affiché montre les unités qu'il a vendu et en quelle quantité.

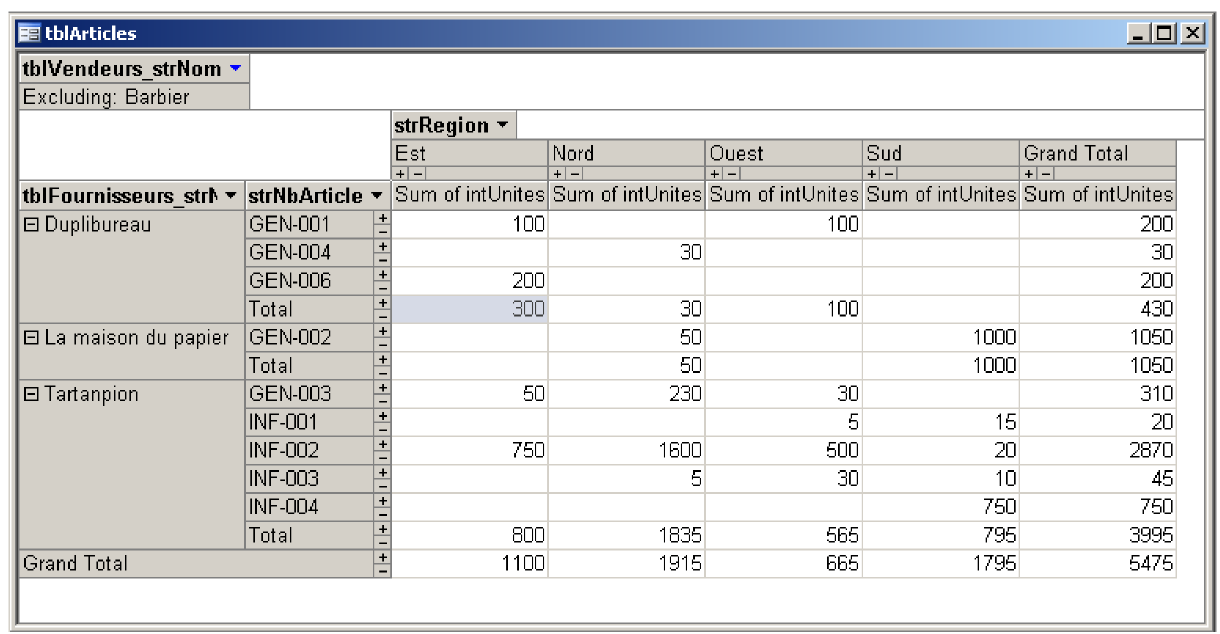
Conseil: utilisez les assistants… en particulier… le menu *Insertion/Graphique* (durée de l'exercice: ½ heure)

Que peut-on en conclure relativement à l'exercice précédent ?

8.9 Tableaux croisés dynamiques

Vous avez certainement remarqué lors de la création de formulaires qu'il vous était proposé de créer un tableau croisé dynamique. Gardons bien en tête que ce genre d'outil sert à faire des rapports d'analyse de synthèse d'où le fait que nous le traitions dans cette section du support.

Ainsi, seuls (car c'est relativement facile), à l'aide de l'assistant de tableau croisé dynamique, créez le TCD suivant (attention nous débordons presque sur un cours MS Excel ici !)


tblArticles

tblVendeurs_strNom
Excluding: Barbier

strRegion

tblFournisseurs_strN	strNbArticle	Est	Nord	Ouest	Sud	Grand Total
		Sum of intUnites	Sum of intUnites	Sum of intUnites	Sum of intUnites	Sum of intUnites
Duplibureau	GEN-001	100		100		200
	GEN-004		30			30
	GEN-006	200				200
	Total	300	30	100		430
La maison du papier	GEN-002		50		1000	1050
	Total		50		1000	1050
Tartanpion	GEN-003	50	230	30		310
	INF-001			5	15	20
	INF-002	750	1600	500	20	2870
	INF-003		5	30	10	45
	INF-004				750	750
	Total	800	1835	565	795	3995
Grand Total		1100	1915	665	1795	5475

Figure 10 Tableau croisé dynamique

Pour plus de précisions sur les TCD nous renvoyons le participant à un cours MS Excel sur le sujet.

Remarques:

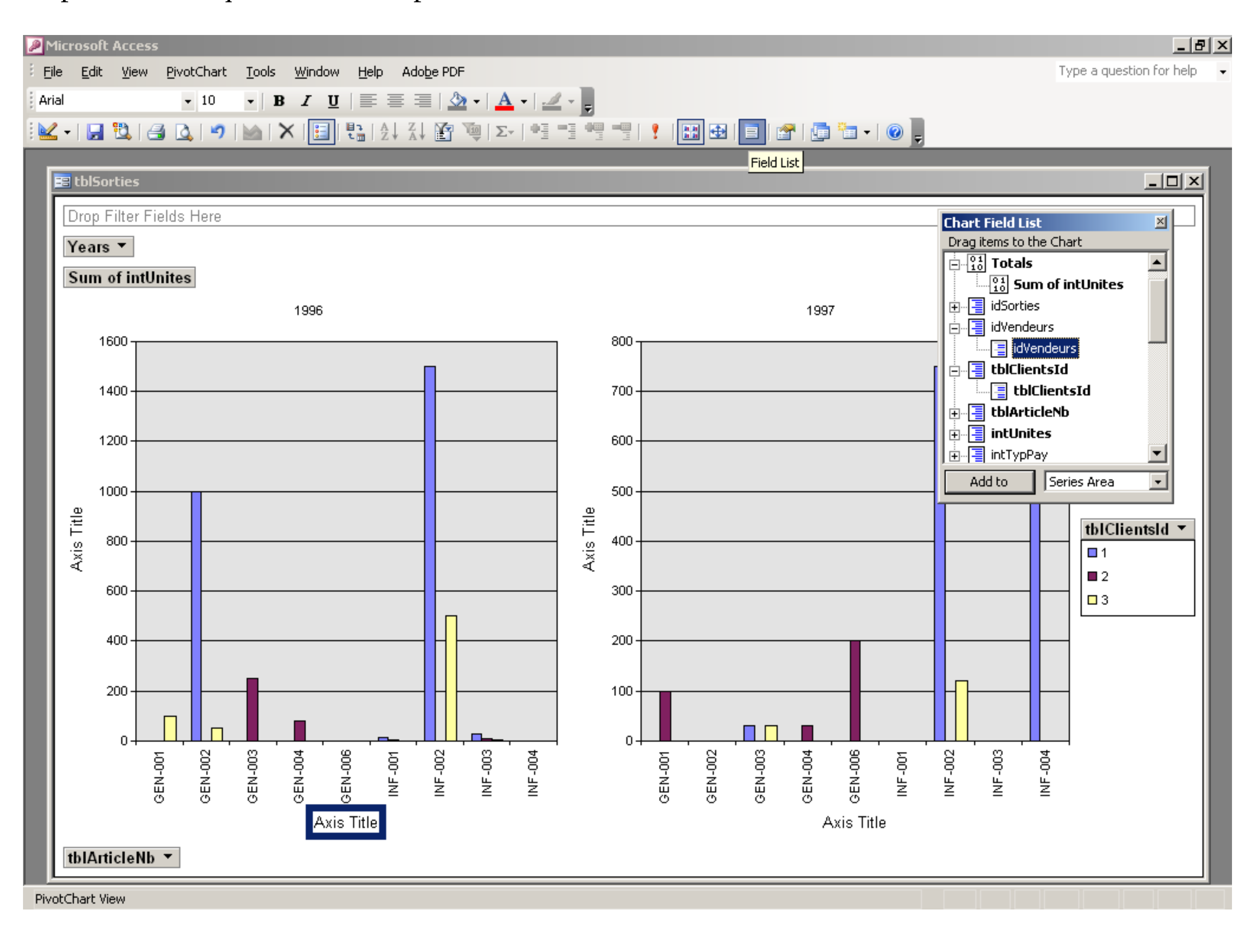
R1. Cet outil est complémentaire aux requêtes croisées dynamiques. Il a aussi un défaut: au-delà de 3000 à 4000 données il faut parfois attendre 15 à 30 secondes avant que le tableau s'affiche. Il vaut mieux dès lors avoir d'autres outils à sa disposition (SQL Server, Oracle avec des Cubes OLAP)

R2. La plupart du temps, les utilisateurs exporteront à l'aide du bouton adéquat, ce TCD vers MS Excel.

8.10 Graphiques croisés dynamiques

Nous pouvons faire aussi des graphiques croisés dynamiques avec un avantage par rapport à Excel… ceux-ci sont beaucoup plus puissants dans MS Access que dans MS Excel.

En prenant la table *tblSorties* et en créant un graphique croisé dynamique en ayant au préalable activé le traçage multiple en activant le bouton nous pouvons après un disposition adéquate des champs obtenir:



Bien sûr, il faut un peu retravailler les couleurs, les légendes et la mise en page mais au moins, nous voyons tout de suite que nous dépassons ce que MS Excel sait faire.

Attention!!! Lors de la création de bouton permettant d'ouvrir un formulaire ou une requête de type tableau ou graphique croisé dynamique, ne pas oublier de forcer le deuxième paramètre de la commande *OpenForm* en indiquant *acFormPivotChart* ou *acFormPivotTable*. Par exemple pour un formulaire avec un GCD:

```
DoCmd.OpenForm stDocName, acFormPivotChart, , stLinkCriteria
```

ou un pour une requête de type GCD:

```
DoCmd.OpenQuery qryName, acViewPivotChart
```

9 Requêtes (complexes avec ou sans VBA)

Commençons tout de suite par un sujet important et qui donne souvent la migraine aux débutants!

Plutôt que de faire un amalgame théorique nous allons de suite faire un exercice afin de voir trivialement de quoi il retourne dans la pratique.

9.1 Requêtes avec jointures

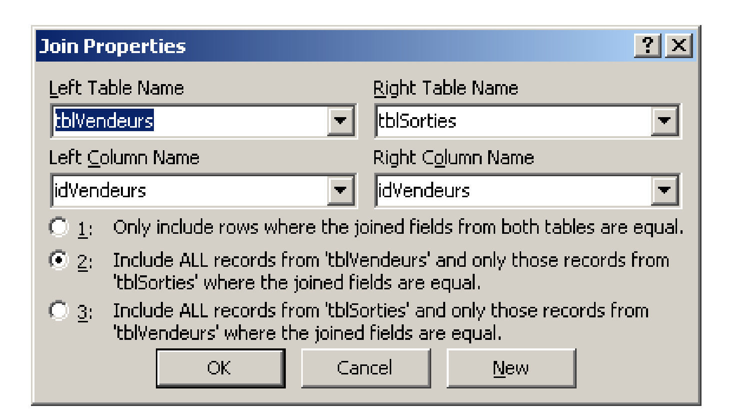
Remarque: la théorie est un peu "floue" à ce sujet (dans le sens où il faut être bien réveillé… mais ce n'est pas dur du tout !!!)

Sous MS Access, comme d'ailleurs dans la plupart des SGBDR, il y a plusieurs types de jointures.

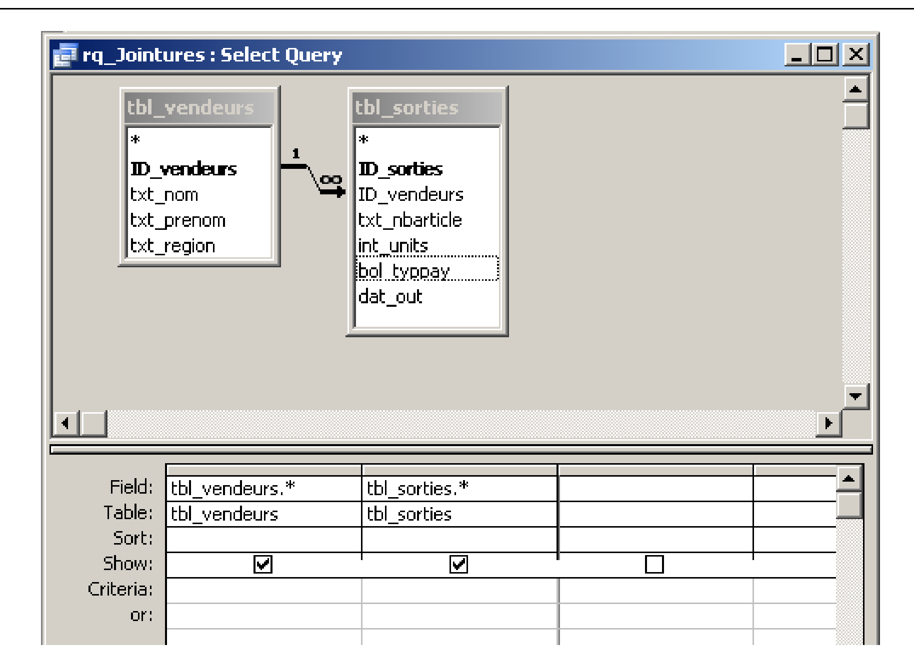
1. La jointure équivalente (dite aussi "interne")
2. La jointure gauche
3. La jointure droite

Ouvrez la table *tblSorties* et faites en sorte que certains vendeurs n'aient rien vendu (c'est juste pour l'exemple). Pour cela, créez de nouveaux vendeurs.

Créez ensuite une nouvelle requête nommée *qryJointures* du type suivant. Pour définir le type de jointure, faites un double clic sur la liaison et la boîte de dialogue suivante apparaîtra:

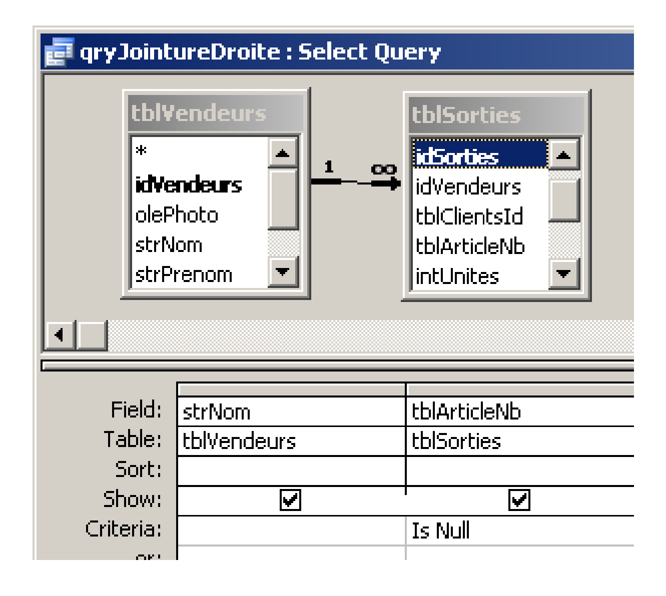


et sélectionnez (2) qui une jointure droite tel que:



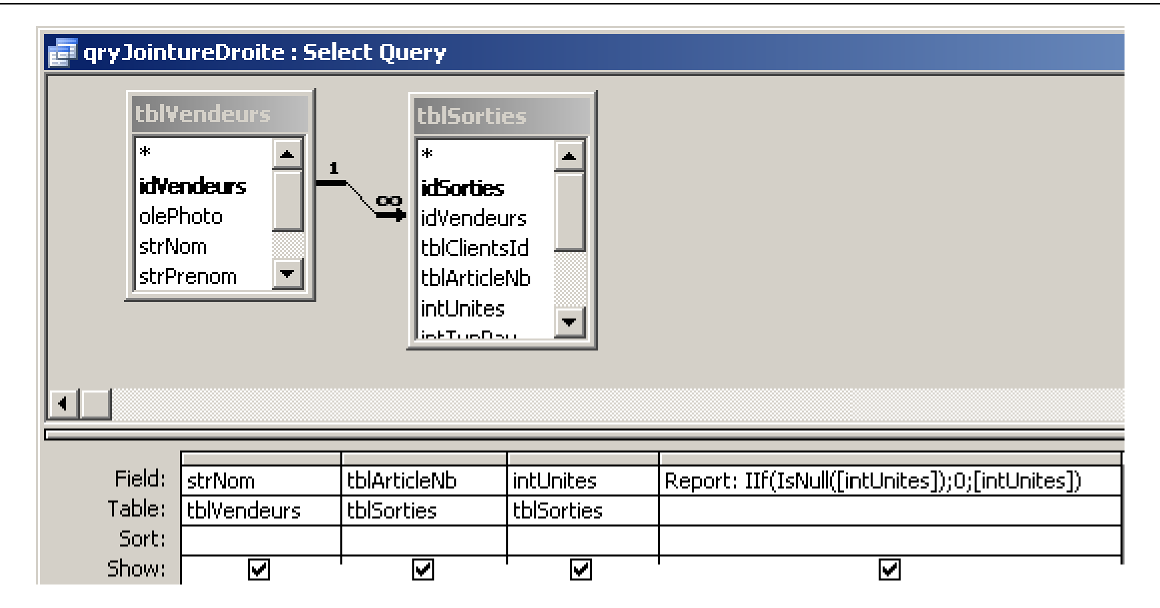
1. Quelle est la différence avec une jointure équivalente ?
2. Que peut-on en conclure pour l'instant ?

Que faut-il ajouter maintenant dans cette requête pour y avoir maintenant seulement la liste des vendeurs qui n'ont rien vendu ?



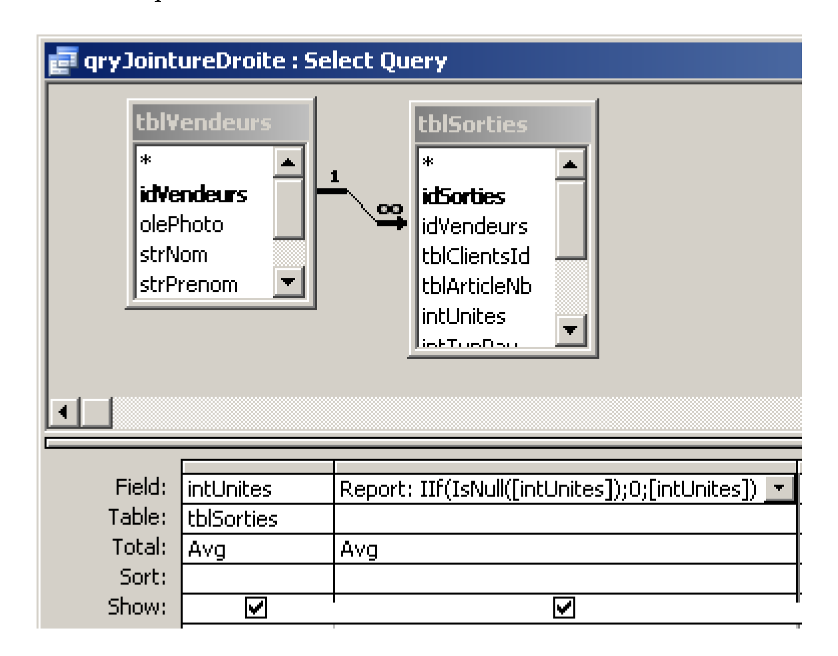
Si vous mettez =0 au lieu de *Is Null* que se passe-t-il ?

Modifiez maintenant la requête afin d'obtenir la chose suivante:

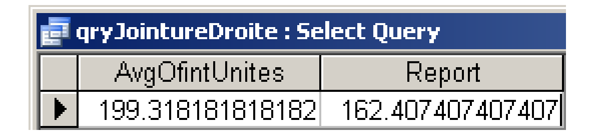


Quelle en est l'utilité ?

Modifiez maintenant la requête de la manière suivante:



Effectivement, le résultat est flagrant !!!:

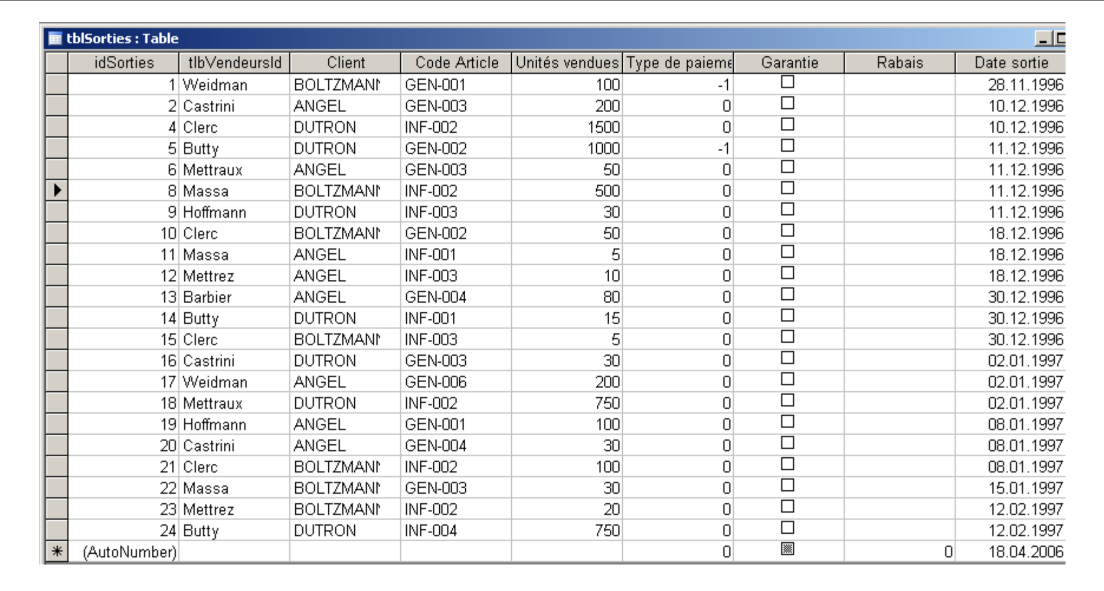

qryJointureDroite : Select Query

AvgOfintUnites	Report
199.318181818182	162.407407407407

Que peut-on en conclure ?

9.2 Requête regroupement par premiers éléments

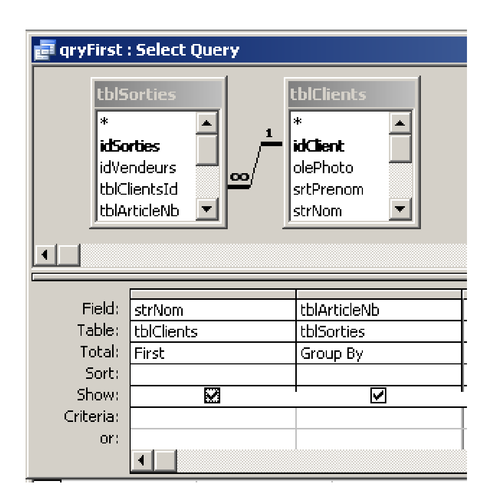
Toujours dans notre base, considérons la table *tblSorties*:


tblSorties : Table

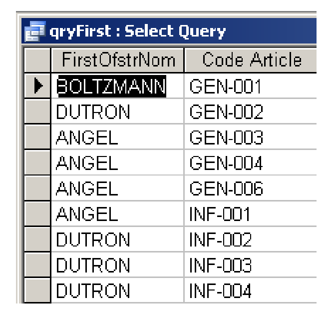
idSorties	tlbVendeursId	Client	Code Article	Unités vendues	Type de paieme	Garantie	Rabais	Date sortie
1	Weidman	BOLTZMANI	GEN-001	100	-1			28.11.1996
2	Castrini	ANGEL	GEN-003	200	0			10.12.1996
4	Clerc	DUTRON	INF-002	1500	0			10.12.1996
5	Butty	DUTRON	GEN-002	1000	-1			11.12.1996
6	Mettraux	ANGEL	GEN-003	50	0			11.12.1996
8	Massa	BOLTZMANI	INF-002	500	0			11.12.1996
9	Hoffmann	DUTRON	INF-003	30	0			11.12.1996
10	Clerc	BOLTZMANI	GEN-002	50	0			18.12.1996
11	Massa	ANGEL	INF-001	5	0			18.12.1996
12	Mettrez	ANGEL	INF-003	10	0			18.12.1996
13	Barbier	ANGEL	GEN-004	80	0			30.12.1996
14	Butty	DUTRON	INF-001	15	0			30.12.1996
15	Clerc	BOLTZMANI	INF-003	5	0			30.12.1996
16	Castrini	DUTRON	GEN-003	30	0			02.01.1997
17	Weidman	ANGEL	GEN-006	200	0			02.01.1997
18	Mettraux	DUTRON	INF-002	750	0			02.01.1997
19	Hoffmann	ANGEL	GEN-001	100	0			08.01.1997
20	Castrini	ANGEL	GEN-004	30	0			08.01.1997
21	Clerc	BOLTZMANI	INF-002	100	0			08.01.1997
22	Massa	BOLTZMANI	GEN-003	30	0			15.01.1997
23	Mettrez	BOLTZMANI	INF-002	20	0			12.02.1997
24	Butty	DUTRON	INF-004	750	0			12.02.1997
(AutoNumber)					0		0	18.04.2006

Nous souhaiterions de cette liste, créer une requête nommée *qryFirst* qui nous indique qui est le client qui a commandé en premier un certain article.

Pour ce faire, il faut créer une requête de ce type:

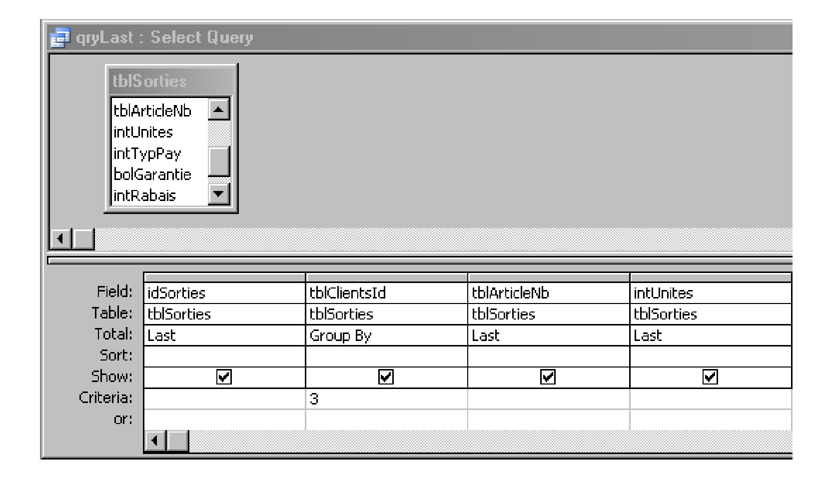


Le résultat sera alors bien:


qryFirst : Select Query

FirstOfstrNom	Code Article
BOLTZMANN	GEN-001
DUTRON	GEN-002
ANGEL	GEN-003
ANGEL	GEN-004
ANGEL	GEN-006
ANGEL	INF-001
DUTRON	INF-002
DUTRON	INF-003
DUTRON	INF-004

Ou afficher la dernière commande effectuée dans une requête nommée *qryLast* par Boltzmann (client n°3):



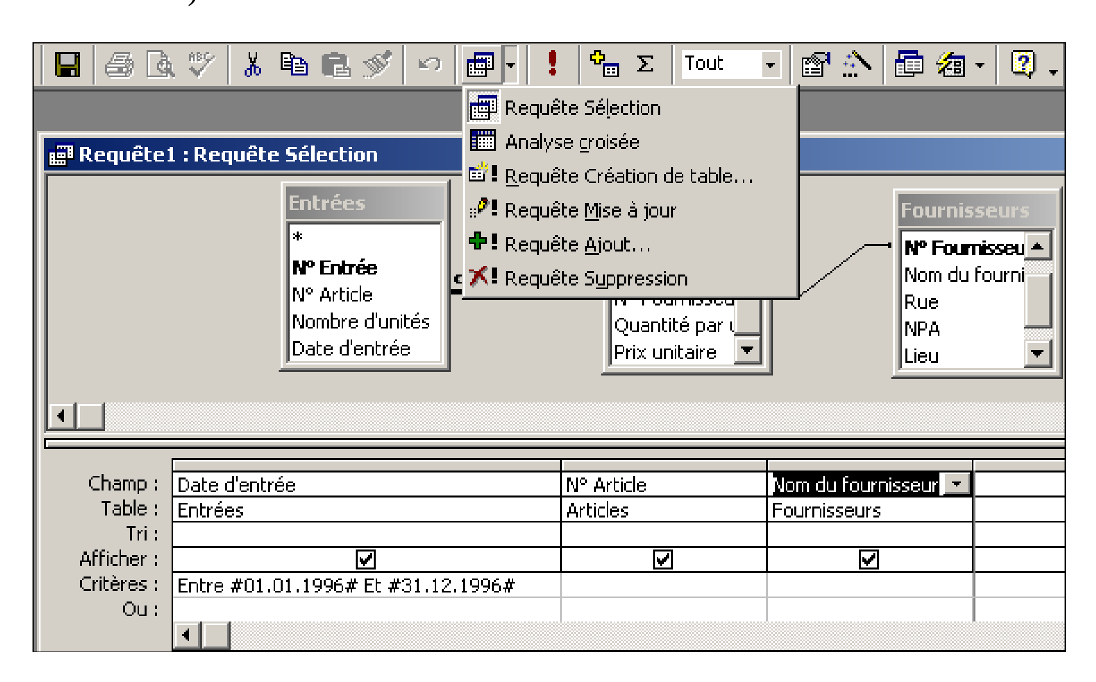
9.3 Requête avec critères

Créez une requête simple avec les propriétés de requêtes suivantes: les jeux d'enregistrements avec la *datDateIn* (Table *tblEntrees*) comprise entre le 01.01.96 et le 31.12.96. Le numéro d'articles (Table *tblArticles*) et le nom du Fournisseur (Table *tblFournisseurs*) doivent y être indiqués.

Exécutez votre requête et observez si elle marche bien comme on l'attendait (on voit que les relations maintenant jouent bien leur rôle aussi dans le cas des requêtes).

Remarque: Il ne peut y avoir d'objets tables et requêtes ayant le même nom dans MS Access!!!

Supposez maintenant qu'un collègue vous demande maintenant de sauvegarder (backuper...) toutes les données d'une certaine année (la solution complète n'est pas dans l'image ci-dessous...) dans un fichier Access contenant uniquement des tables de sauvegarde. Il vous faudra exécuter la requête en sorte qu'elle crée une table physique réelle et aussi une requête qui nettoyer les anciennes données de la table active (faites une copie du fichier de cours avant de tester cela).



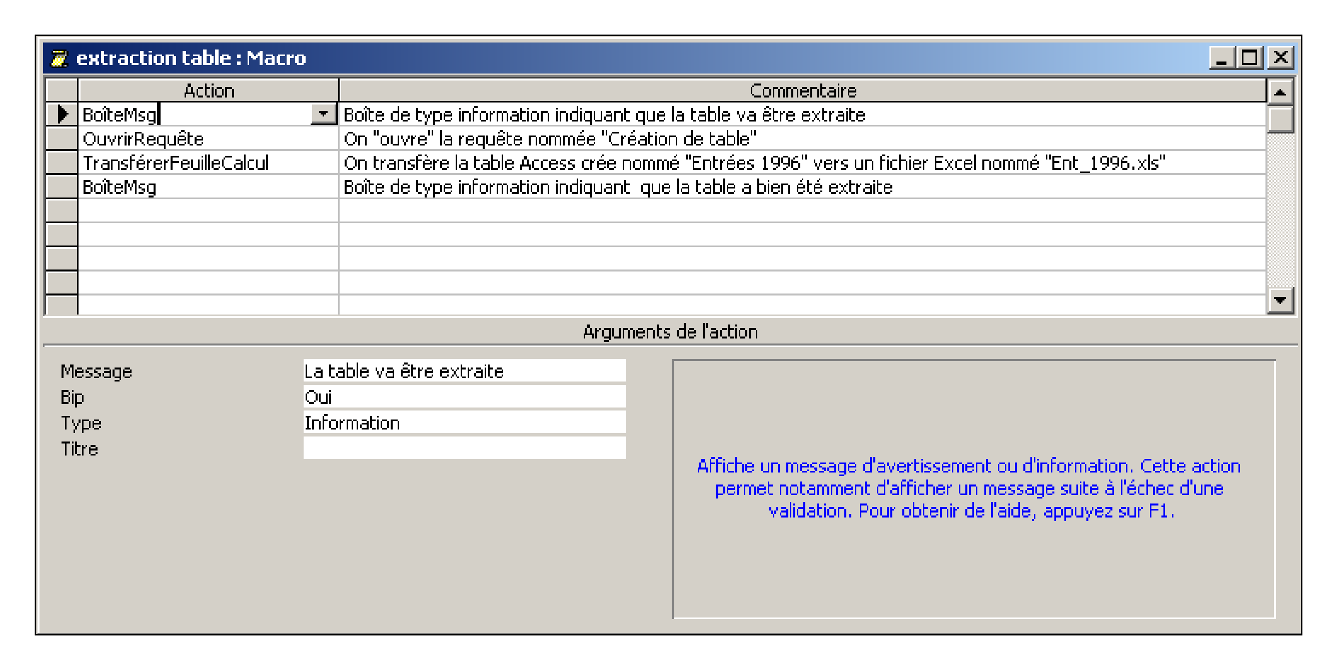
Pour cela, dans le bouton menu à gauche du bouton d'exécution de la requête, il y a une option nommée *Requête Création de table..*. Sélectionnez cette option et choisissez comme nom de la nouvelle table: *tblEntrees1996*. Ceci fait, exécutez la requête, contrôlez que la nouvelle table ait bien été créé dans le fichier Access préparé à cet effet.

Ce type de manipulation est fréquent dans les entreprises car les employés ont souvent besoin d'analyser les données pour leur compte dans un autre fichier Access (on peut faire de même avec MS Excel!) Or, on ne peut leur demander d'effectuer les opérations que nous venons de faire (ils n'ont ni le niveau de connaissance ni les droits).

Il faudrait donc créer un petit programme (nommé "Macro" par Microsoft) qui permet à partir d'un bouton sur un formulaire (le formulaire d'accueil de la base typiquement) d'effectuer toutes ces opérations automatiquement.

La création et l'enregistrement d'une Macro est une chose extrêmement simple mais encore faut-il savoir quelles actions utiliser (les combinaisons possibles sont suffisamment grandes pour passer plusieurs années à les étudier…).

Nous allons faire cela en tant qu'exercice (voir ci-dessous une macro similaire à celle dont nous avons besoin, à vous de l'adapter!).



Remarque: Quand vous effectuez des requêtes simples (et non des requêtes d'action), il est bien sûr possible d'imprimer la requête (action: *Imprimer*) puisqu'elle va s'ouvrir automatiquement et de la fermer de suite après (action: *Fermer*).

9.4 Requête d'analyse de fréquence (contingence)

Souvent les gens souhaitent faire des statistiques de fréquence pour savoir quelle quantité de choses se trouve dans un nombre d'intervalles donnés.

Faisons un exemple avec la table *tblSorties*. Nous souhaiterions savoir combien de ventes il y a eu (tous produits confondus) entre 0-50 unités, 51-100 unités, 101-150 unités, 151-250 unités, 251-500 unités, 501-1000 unités, 1001-2000 unités.

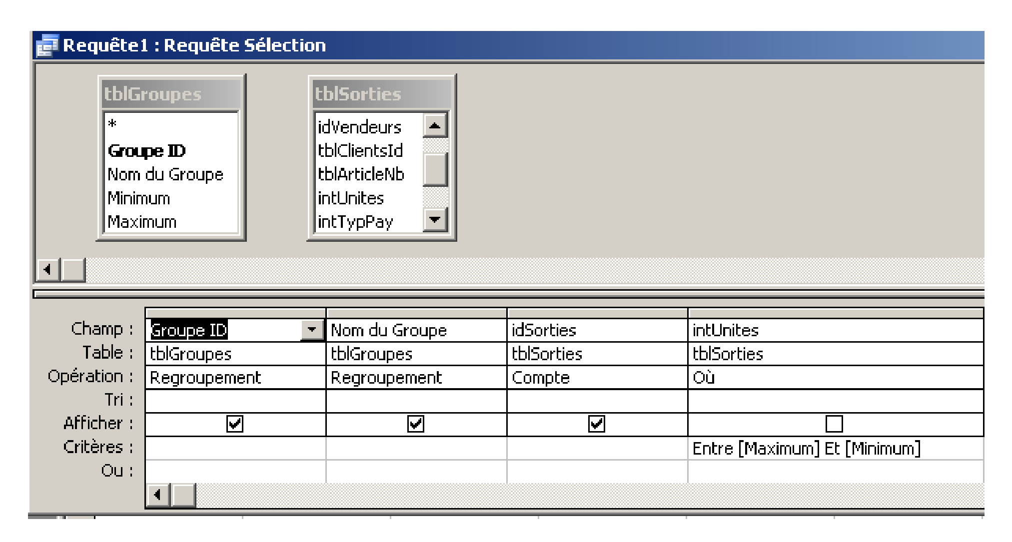
Pour faire cela, créez d'abord une table nommée *tblGroupes* avec la structure suivante:

tblGroupes : Table

Groupe ID	Nom du Groupe	Minimum	Maximum
1	Sous 50	0	50
2	51 à 100	51	100
3	101 à 150	101	150
4	151 à 250	151	250
5	251 à 500	251	500
6	501 à 1000	501	1000
7	1001 à 2000	1001	2000
(NuméroAuto)		0	0

Attention à choisir correctement l'unité de fin de la valeur minimum: …01 ou ….,01

Créez ensuite la requête suivante *qryFrequences* sans mettre de liaison entre les deux tables:



En exécutant la requête, nous obtiendrons alors:

Requête1 : Requête Sélection

Groupe ID	Nom du Groupe	CompteDeidSorties
1	Sous 50	13
2	51 à 100	4
4	151 à 250	2
5	251 à 500	1
6	501 à 1000	3
7	1001 à 2000	1

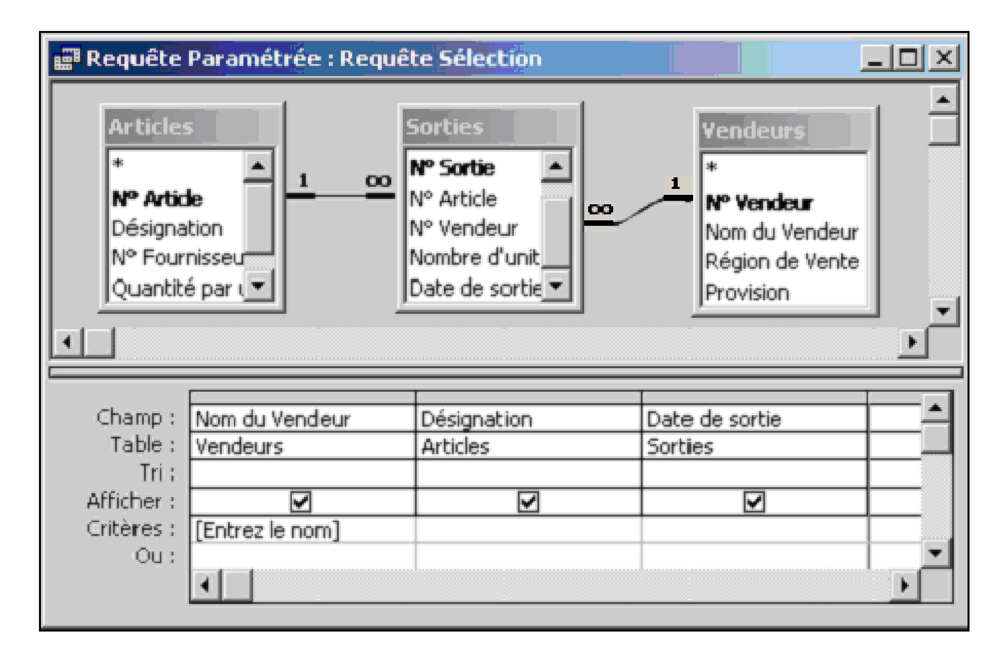
9.5 Requête paramétrée

Dans la même base créez une requête qui établit la liste de tous les articles avec les dates de sorties, qui ont été vendu par un vendeur donné. Le nom du collaborateur doit pouvoir être fixé lors de l'exécution de la requête.

Il s'agit d'une requête dite "requête paramétrée" sans aucune spécification. Voilà comment procéder:

1. Créer une nouvelle requête

2. Glisser les tables dans la requête



3. Dans le champ critères taper: [Entrez le nom] (les termes entre crochets ne doivent pas contenir le nom d'un champ existant dans la table utilisée!!!)

4. Exécuter la requête avec le nom *Castrini*

5. Enregistrez la requête sous le nom *qryParametree*

Problème: vous voyez que lorsque l'utilisateur ne saisit rien dans le paramètre de la requête, rien ne s'affiche (ce qui peut être vu comme fort ennuyeux). Pour remédier à cela, il suffit d'écrire la chose suivante:

[Nom du Vendeur:] Or Is Null

Sinon avec un peu d'imagination voici le genre de paramètres (exemples importants et à tester!!!) que vous pouvez définir (concepts similaires aux filtres vus au début du cours):

Like [tapez ici] & ""*

Like "" & [tapez ici] & "*"*

Year([ChampAvecDate])=[Saisissez une année]

Month([ChampAvecDate])=[Saisissez un mois entre 1 et 12]

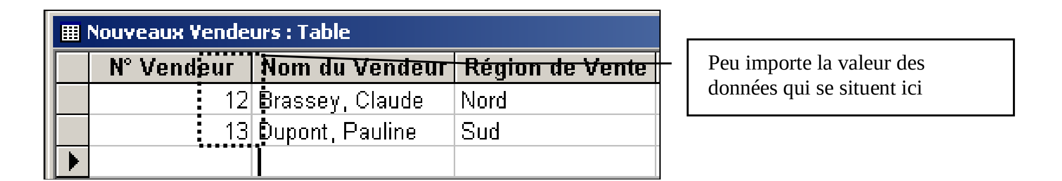
Month([ChampAvecDate])=[Saisir] And Year([ChampAvecDate])=[Saisir]

>Date() – [Données depuis X jours]

la syntaxe ci-dessus change si vous avez activé l'ANSI 92 dans Access (le *Like* devient *ALike* et le "*" devient "%").

9.6 Requête d'ajout

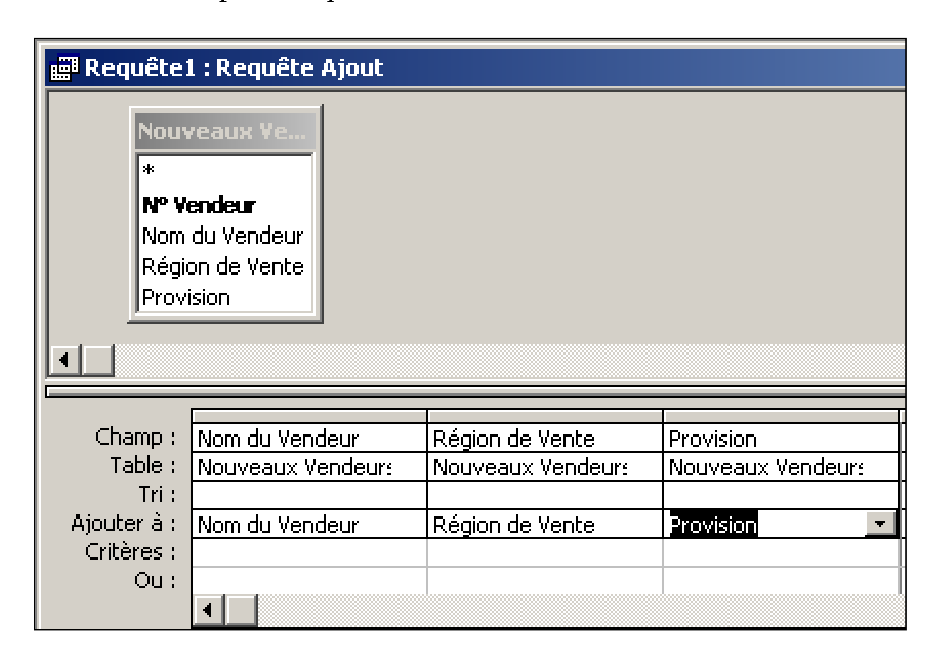
Dans la même base créez la table *tlbNouveauxVendeurs* suivante (si elle ne se trouve pas déjà dans la base de données) ayant les mêmes propriétés de masque de saisie que la table *tblVendeurs*:



Créez une nouvelle requête (en mode création) et redéfinissez la requête comme étant une *qryAjoutVendeurs…*

Dans le champ *Ajouter à*, sélectionnez la table *tblVendeurs*.

Définissez ensuite les champs de requêtes ainsi:



Exécutez la requête et allez regarder dans la table *tblVendeurs*

Ce genre de requête d'ajout est très pratique si vous devez insérer dans votre base les données d'un utilisateur (ou plusieurs) travaillant dans MS Excel (ce qui arrive presque toujours) ou lorsque vous faites de gros changements dans la structure de votre base et que vous devez déplacer de grosses quantités de donnés d'une ancienne structure de table à une nouvelle.

Une bonne idée en ce qui concerne les requêtes d'ajout est de mettre dans la table cible une colonne "*datDateAjout*" avec une valeur par défaut du type *Now*( ). Ainsi, vous avez une traçabilité de quand ont été ajout les données afin éventuellement de pouvoir plus facilement les supprimer.

9.7 Requête de suppression

Créez maintenant deux "*Requête de suppression*" (en ayant sélectionné au préalable la table *Vendeurs*).

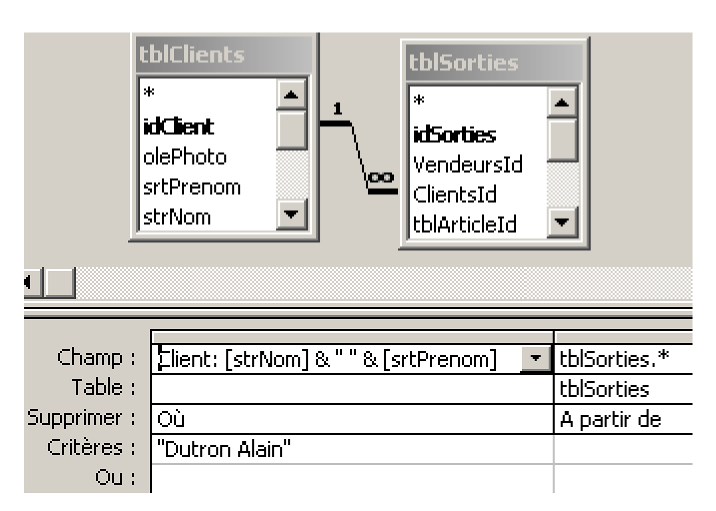
Cette dernière s'utilise comme une requête classique où l'on doit définir les critères qui détermineront quels enregistrements seront supprimés.

1. Dans le premier exercice, avec votre formateur, supprimez les enregistrements ajoutés précédemment à la table *tblVendeurs* et enregistrez la requête sous le nom de *qrySuppression*.

   Ce type de requête est très utile dans le genre de cas suivant:

   Légalement et suivant l'activité, les entreprises peuvent tous les cinq ou dix ans (en fonction du type de documents) se débarrasser de toutes leurs archives ou historique. Dans le cas d'une base de données MS Access, il est possible pour alléger la taille mémoire de la base, de faire une copie de cette dernière tous les cinq ans et d'utiliser ensuite une nouvelle base où tous les enregistrements antérieurs à ces temps sont supprimés.

2. Créez une deuxième requête de suppression nommée *qrySupprSort* qui supprime toutes les sorties pour le client *Dutron Alain*.



9.8 Requête d'analyse croisée (avec assistant)

Créez maintenant une requête d'analyse croisée avec l'assistant qui:

1. Groupe par vendeur les articles

2. Affiche par vendeur et par article les ventes d'unités pour chaque client

Le résultat devrait ressembler à la chose suivante:

tblSorties_Crosstab : Crosstab Query

Vendeurs	Code Article	Total Of intUnites	1	2	3
1	GEN-001	50			50
1	GEN-006	200		200	
2	GEN-002	100	100		
2	INF-001	15	15		
2	INF-004	750	750		
3	GEN-002	50			50
3	INF-002	1600	1500		100
3	INF-003	5			5
4	GEN-003	40	30	10	
4	GEN-004	30		30	
6	GEN-003	30			30
6	INF-001	5		5	
6	INF-002	500			500
7	INF-002	20			20
7	INF-003	10		10	
8	GEN-004	80		80	
9	GEN-003	20		20	
9	INF-002	750	750		
10	GEN-001	100		100	
10	INF-003	30	30		

Nous remarquons tout de suite le problème inhérent au fait d'avoir fait des relations avec le *idClient* au lieu *strNom*. Il n'est pas possible ici de corriger ce problème sans faire appel aux jointures.

9.9 Requête d'analyse croisée (sans assistant)

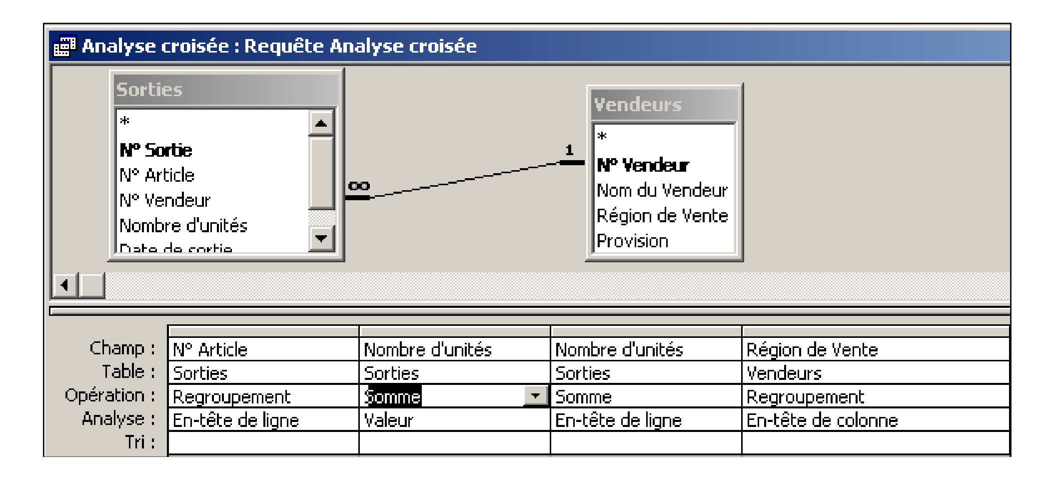
Essayez maintenant de créer manuellement (sans l'assistant) une autre requête d'analyse croisée qui (petit truc…: pensez à MS Excel pour le vocabulaire…):

1. Groupe les articles
2. Effectue la somme des unités vendues, et par région comme ci-dessous:

Analyse croisée : Requête Analyse croisée

N° Article	SommeDeNombre d'unités	Est	Nord	Ouest	Sud
GEN-001	150	100			50
GEN-002	150		50	100	
GEN-003	90	20	40	30	
GEN-004	110			30	80
GEN-005	50	10	40		
GEN-006	200		200		
INF-001	20	20			
INF-002	2870	2250	100	20	500
INF-003	45	35		10	
INF-004	750	750			

Remarque: au besoin, si vous avez des difficultés, créez une requête en mode création et redéfinissez-la comme "*Requête d'analyse croisée*" et définissez les champs comme ci-dessous:



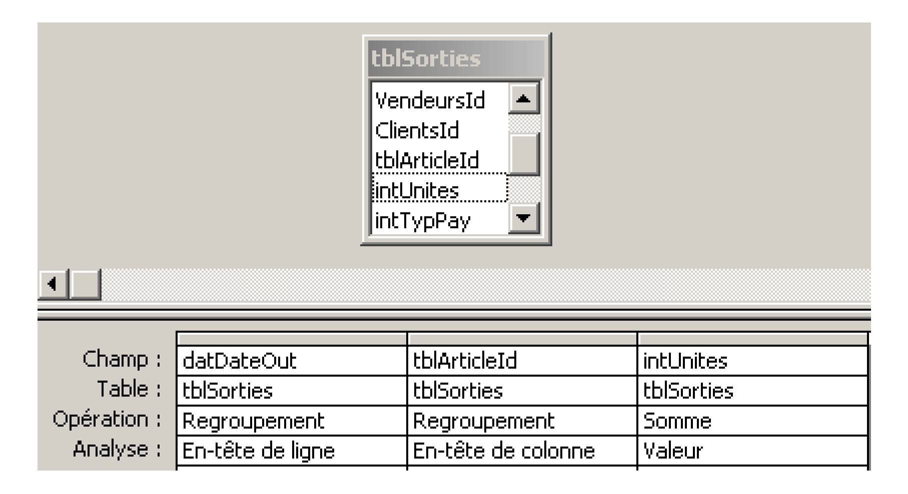
1. Exécutez la requête

2. Enregistrez la sous le nom *qryComplexCrossTab*

Si vous allez voir dans la table *tblVendeurs*, vous verrez que le résultat est juste.

9.10 Requête d'analyse croisée temporelle

Nous souhaitons dans une requête d'analyse croisée nommée *qryAnalTime* afficher la synthèse suivante, de la table *tblSorties*:



dont le résultat est:

Date sortie	GEN-001	GEN-002	GEN-003	GEN-004	GEN-006	INF-001	INF-002	INF-003	INF-004
28.11.1996	50								
10.12.1996			10				1500		
11.12.1996		100	20				500	30	
18.12.1996		50				5		10	
30.12.1996				80		15		5	
02.01.1997			30		200		750		
08.01.1997	100			30			100		
15.01.1997			30						
12.02.1997							20		750

Maintenant, nous souhaiterions regrouper les données par paquets de 1 mois. Pour ce faire, saisissez la formule suivante dans le champ *datDateOut*:

Dates: Format([datDateOut];"aaaa/m")
Regroupement
En-tête de ligne

Cela donnera alors le résultat suivant:

Dates	GEN-001	GEN-002	GEN-003	GEN-004	GEN-006	INF-001	INF-002	INF-003	INF-004
1996/11	50								
1996/12		150	30	80		20	2000	45	
1997/1	100		60	30	200		850		
1997/2							20		750

Cette syntaxe fonctionne pour (attention pour les codes si vous travaillez sur une version anglophone):

1. Les regroupements par jours: Format([datDateOut];"aaaa/j")
2. Les regroupements par semaines: Format([datDateOut];"aaaa/ww")
3. Les regroupements par mois: Format([datDateOut];"aaaa/m")
4. Les regroupements par trimestres: Format([datDateOut];"aaaa/t")
5. Les regroupements par années: Format([datDateOut];"aaaa")

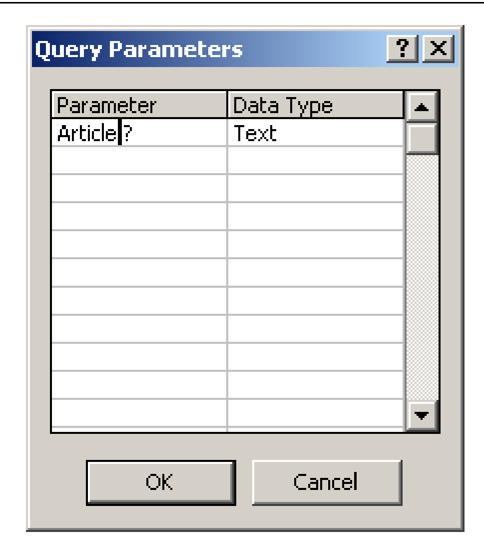
9.11 Requête d'analyse croisée paramétrée

Copiez la requête faite précédemment sous le nom *qryCompCrossTabPar* et dans la colonne *N° Article* (*strCodeArticle*) ajoutez les paramètres (requête croisée paramétrée):

[Article ?] OR is Not Null

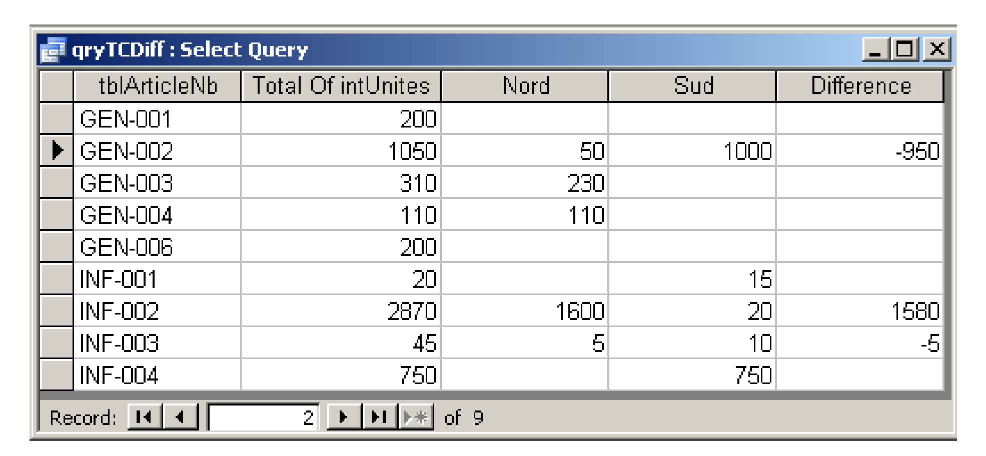
Exécutez la requête ! Pourquoi ne fonctionne-elle plus ?

Au fait, il s'agit d'un cas particulier des requêtes croisées. Il faut définir les champs paramétrés dans une zone spéciale en allant dans le menu *Requête / paramétrée* et y **copier** les paramètres sans crochets (s'il s'agit de paramètres simples) ou avec crochets (s'il s'agit de paramètres références à un formulaire) en spécifiant bien le type de données à utiliser **ET** en laissant toujours la question d'origine dans le champ *strCodeArticle*:



La requête fonctionne maintenant. N'oubliez jamais cela !!!

Comme exercice, créez une requête *qryDiffNordSud* de sélection basée sur la requête précédent afin d'avoir le résultat suivant:


qryTCDiff : Select Query

tblArticleNb	Total Of intUnites	Nord	Sud	Difference
GEN-001	200			
GEN-002	1050	50	1000	-950
GEN-003	310	230		
GEN-004	110	110		
GEN-006	200			
INF-001	20		15	
INF-002	2870	1600	20	1580
INF-003	45	5	10	-5
INF-004	750		750	

Record: 2 of 9

9.12 Requête (trouver les) doublons

Dans la même base, créer une requête pour trouver les doublons de la table *tblEntree*.

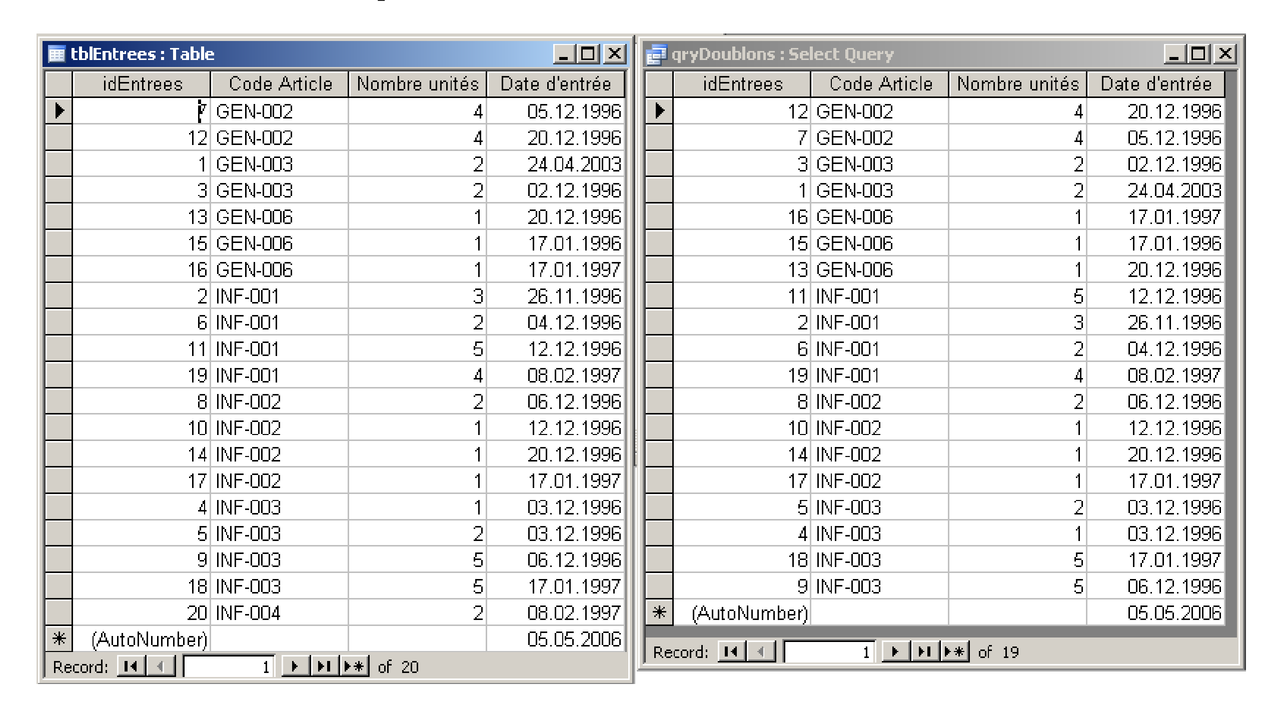
Méthode:

1. Créer une requête de recherche des doublons avec l'assistant
2. Avec l'assistant choisir la table *tblEntree* et ensuite le champ *strArticleNb*
3. Enregistrer la requête sous *qryDoublons*

Attention!!! Cette requête n'a pas pour but d'effacer les enregistrements (lignes) à double mais seulement de chercher les champs qui sont doubles et d'afficher ceux-ci.

A tort, certaines personnes l'utilisent pour créer des listes uniques en la bricolant en Mode Création. C'est totalement faux et ce n'est pas son usage (il est alors préférable d'utiliser une requêtes faisant un regroupement ou une requête d'unicité).

Ainsi, le résultat de l'exemple susmentionné sera:



tblEntrees : Table

idEntrees	Code Article	Nombre unités	Date d'entrée
7	GEN-002	4	05.12.1996
12	GEN-002	4	20.12.1996
1	GEN-003	2	24.04.2003
3	GEN-003	2	02.12.1996
13	GEN-006	1	20.12.1996
15	GEN-006	1	17.01.1996
16	GEN-006	1	17.01.1997
2	INF-001	3	26.11.1996
6	INF-001	2	04.12.1996
11	INF-001	5	12.12.1996
19	INF-001	4	08.02.1997
8	INF-002	2	06.12.1996
10	INF-002	1	12.12.1996
14	INF-002	1	20.12.1996
17	INF-002	1	17.01.1997
4	INF-003	1	03.12.1996
5	INF-003	2	03.12.1996
9	INF-003	5	06.12.1996
18	INF-003	5	17.01.1997
20	INF-004	2	08.02.1997
(AutoNumber)			05.05.2006

Record: 1 of 20

qryDoublons : Select Query

idEntrees	Code Article	Nombre unités	Date d'entrée
12	GEN-002	4	20.12.1996
7	GEN-002	4	05.12.1996
3	GEN-003	2	02.12.1996
1	GEN-003	2	24.04.2003
16	GEN-006	1	17.01.1997
15	GEN-006	1	17.01.1996
13	GEN-006	1	20.12.1996
11	INF-001	5	12.12.1996
2	INF-001	3	26.11.1996
6	INF-001	2	04.12.1996
19	INF-001	4	08.02.1997
8	INF-002	2	06.12.1996
10	INF-002	1	12.12.1996
14	INF-002	1	20.12.1996
17	INF-002	1	17.01.1997
5	INF-003	2	03.12.1996
4	INF-003	1	03.12.1996
18	INF-003	5	17.01.1997
9	INF-003	5	06.12.1996
(AutoNumber)			05.05.2006

Record: 1 of 19

Nous voyons bien que la table de droite, qui n'est d'autre que la requête, n'affiche pas un élément de la table de droite qui est le seul élément dont le code d'article n'est pas à double dans la table *tblEntrees*; soit *INF-004*.

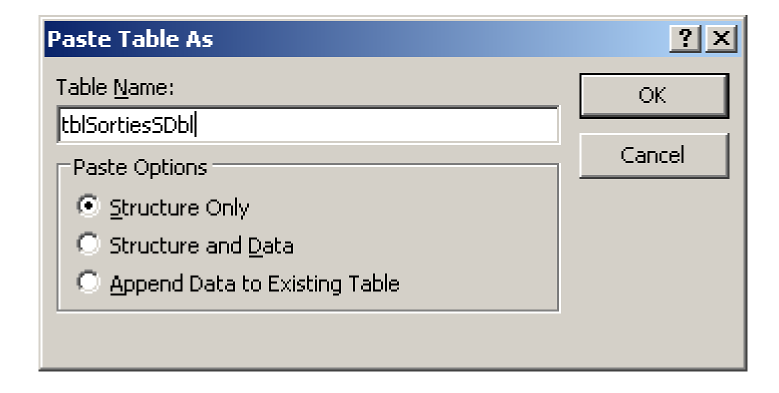
9.13 Requête de suppression des doublons d'enregistrements

C'est le cas le plus fréquemment demandé par les utilisateurs de MS Access. Pour tester cet exemple, nous demanderons au lecteur d'insérer exprès des doublons d'enregistrements dans la table *tblSorties* (quelques copier-coller suffiront).

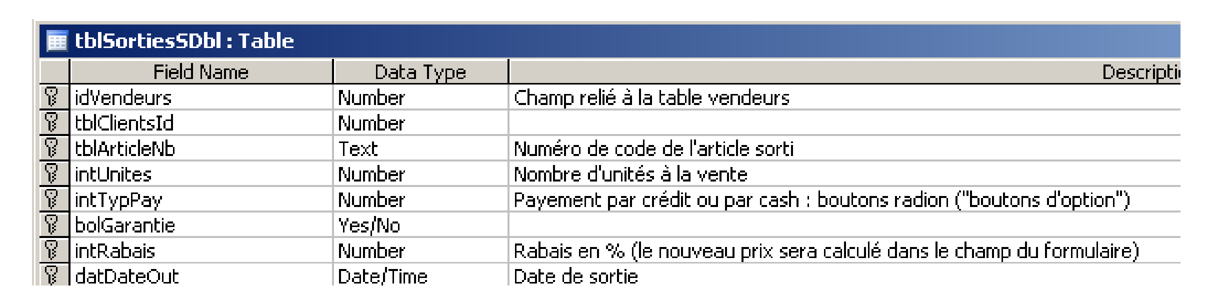
9.13.1 Première approche: non-destructive sur la table originale

Voici la manipulation à faire pour créer ce genre de requêtes:

1. Faites une copie de la structure de votre table *tblSorties*



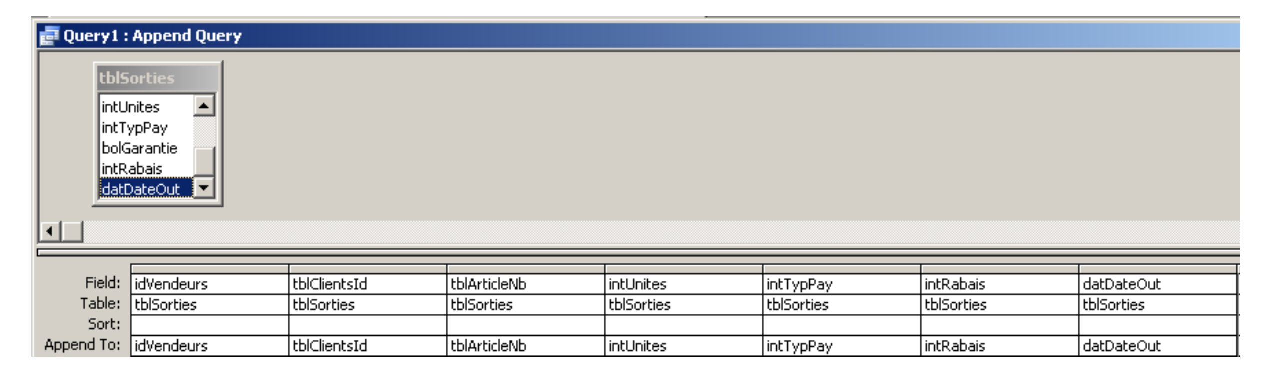
2. Effacez-y le champ de clé primaire *idSorties* et mettez des clés primaires multiples sur tous les autres champs (normalement on fait plutôt cela avec un index sans doublons):



tblSortiesSDbl : Table

Field Name	Data Type	Descripti
idVendeurs	Number	Champ relié à la table vendeurs
tblClientsId	Number	
tblArticleNb	Text	Numéro de code de l'article sorti
intUnites	Number	Nombre d'unités à la vente
intTypPay	Number	Payement par crédit ou par cash : boutons radion ("boutons d'option")
bolGarantie	Yes/No	
intRabais	Number	Rabais en % (le nouveau prix sera calculé dans le champ du formulaire)
datDateOut	Date/Time	Date de sortie

3. Créez ensuite une requête d'ajout qui aura pour objectif de déplacer tous les champs de la table *tblSorties* vers la table *tblSortiesSDbl*



Field:	idVendeurs	tblClientsId	tblArticleNb	intUnites	intTypPay	intRabais	datDateOut
Table:	tblSorties	tblSorties	tblSorties	tblSorties	tblSorties	tblSorties	tblSorties
Sort:							
Append To:	idVendeurs	tblClientsId	tblArticleNb	intUnites	intTypPay	intRabais	datDateOut

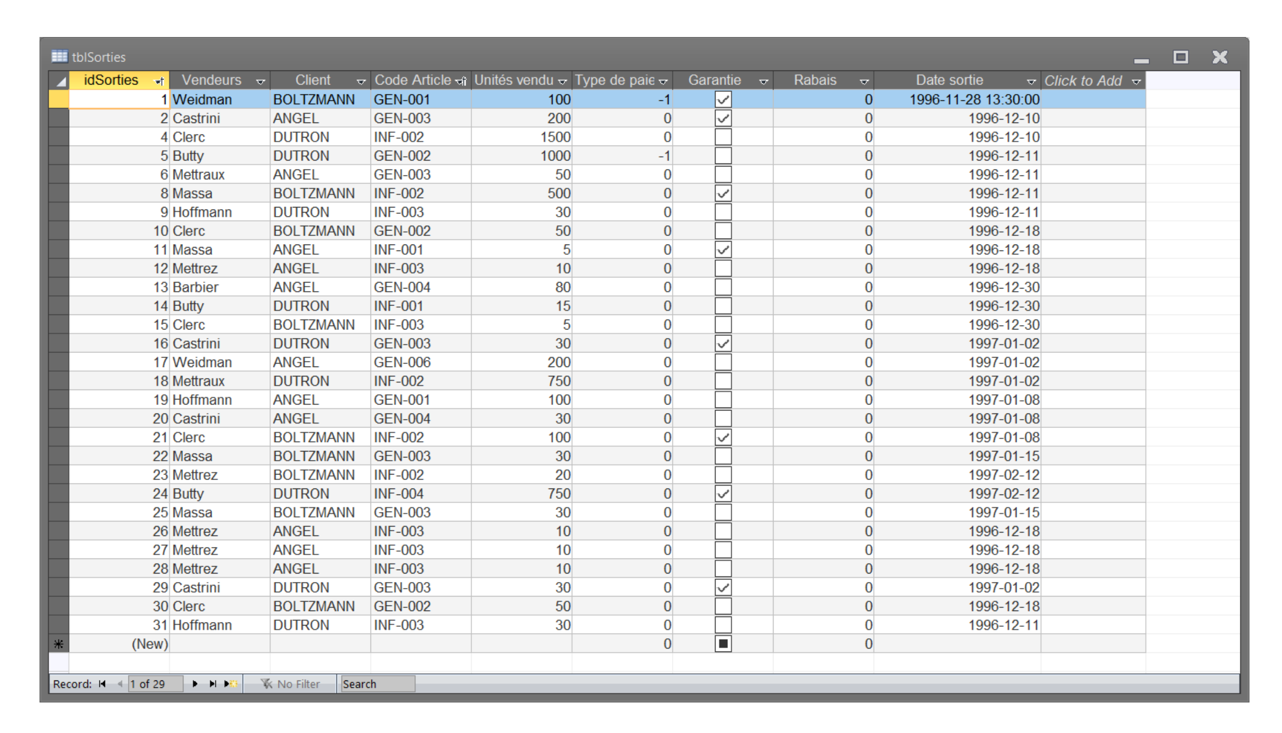
Il suffit ensuite d'exécuter la requête pour avoir dans la table *tblSortiesSDbl* tous les enregistrements sans doublons.

Attention!!! Cette technique qui suppose l'utilisation de clés primaires multiples implique que l'ensemble des enregistrements ont des champs non vides et que le nombre de clés primaires ne dépasse pas 10.

Attention!!! Cette technique ne prendra pas les lignes dont les champs comprenant les clés primaires dans la table cible sont vierges.

9.13.2 Deuxième approche: Destructive sur la table d'origine

Considérons la table suivante:



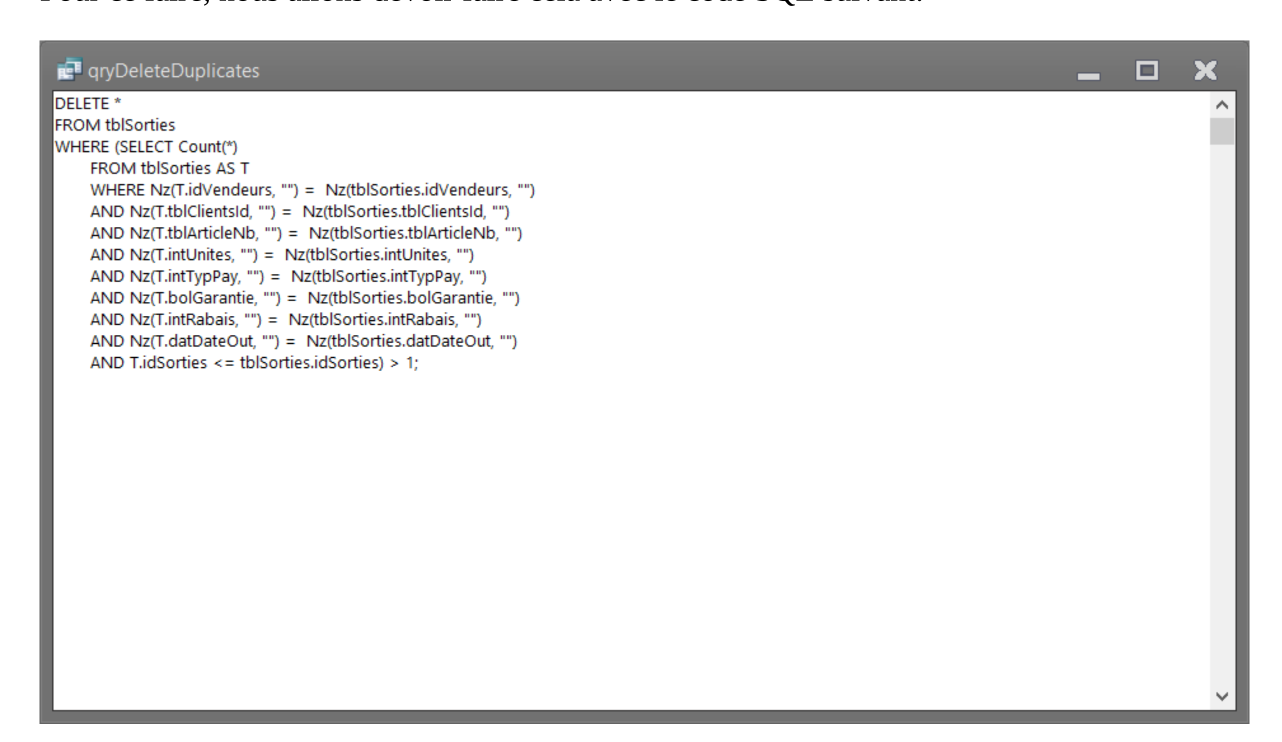
tblSorties

idSorties	Vendeurs	Client	Code Article	Unités vendu	Type de paie	Garantie	Rabais	Date sortie	Click to Add
1	Weidman	BOLTZMANN	GEN-001	100	-1	☑	0	1996-11-28 13:30:00	
2	Castrini	ANGEL	GEN-003	200	0	☑	0	1996-12-10	
4	Clerc	DUTRON	INF-002	1500	0	☐	0	1996-12-10	
5	Butty	DUTRON	GEN-002	1000	-1	☐	0	1996-12-11	
6	Mettraux	ANGEL	GEN-003	50	0	☐	0	1996-12-11	
8	Massa	BOLTZMANN	INF-002	500	0	☑	0	1996-12-11	
9	Hoffmann	DUTRON	INF-003	30	0	☐	0	1996-12-11	
10	Clerc	BOLTZMANN	GEN-002	50	0	☐	0	1996-12-18	
11	Massa	ANGEL	INF-001	5	0	☑	0	1996-12-18	
12	Mettrez	ANGEL	INF-003	10	0	☐	0	1996-12-18	
13	Barbier	ANGEL	GEN-004	80	0	☐	0	1996-12-30	
14	Butty	DUTRON	INF-001	15	0	☐	0	1996-12-30	
15	Clerc	BOLTZMANN	INF-003	5	0	☐	0	1996-12-30	
16	Castrini	DUTRON	GEN-003	30	0	☑	0	1997-01-02	
17	Weidman	ANGEL	GEN-006	200	0	☐	0	1997-01-02	
18	Mettraux	DUTRON	INF-002	750	0	☐	0	1997-01-02	
19	Hoffmann	ANGEL	GEN-001	100	0	☐	0	1997-01-08	
20	Castrini	ANGEL	GEN-004	30	0	☐	0	1997-01-08	
21	Clerc	BOLTZMANN	INF-002	100	0	☑	0	1997-01-08	
22	Massa	BOLTZMANN	GEN-003	30	0	☐	0	1997-01-15	
23	Mettrez	BOLTZMANN	INF-002	20	0	☐	0	1997-02-12	
24	Butty	DUTRON	INF-004	750	0	☑	0	1997-02-12	
25	Massa	BOLTZMANN	GEN-003	30	0	☐	0	1997-01-15	
26	Mettrez	ANGEL	INF-003	10	0	☐	0	1996-12-18	
27	Mettrez	ANGEL	INF-003	10	0	☐	0	1996-12-18	
28	Mettrez	ANGEL	INF-003	10	0	☐	0	1996-12-18	
29	Castrini	DUTRON	GEN-003	30	0	☑	0	1997-01-02	
30	Clerc	BOLTZMANN	GEN-002	50	0	☐	0	1996-12-18	
31	Hoffmann	DUTRON	INF-003	30	0	☐	0	1996-12-11	
(New)					0	■	0		

Record: 1 of 29 No Filter Search

Nous y voyons de toute évidence des doublons Nous souhaiterions supprimer tous les doublons en les définissants sur tous les champs excepté la colonne ID.

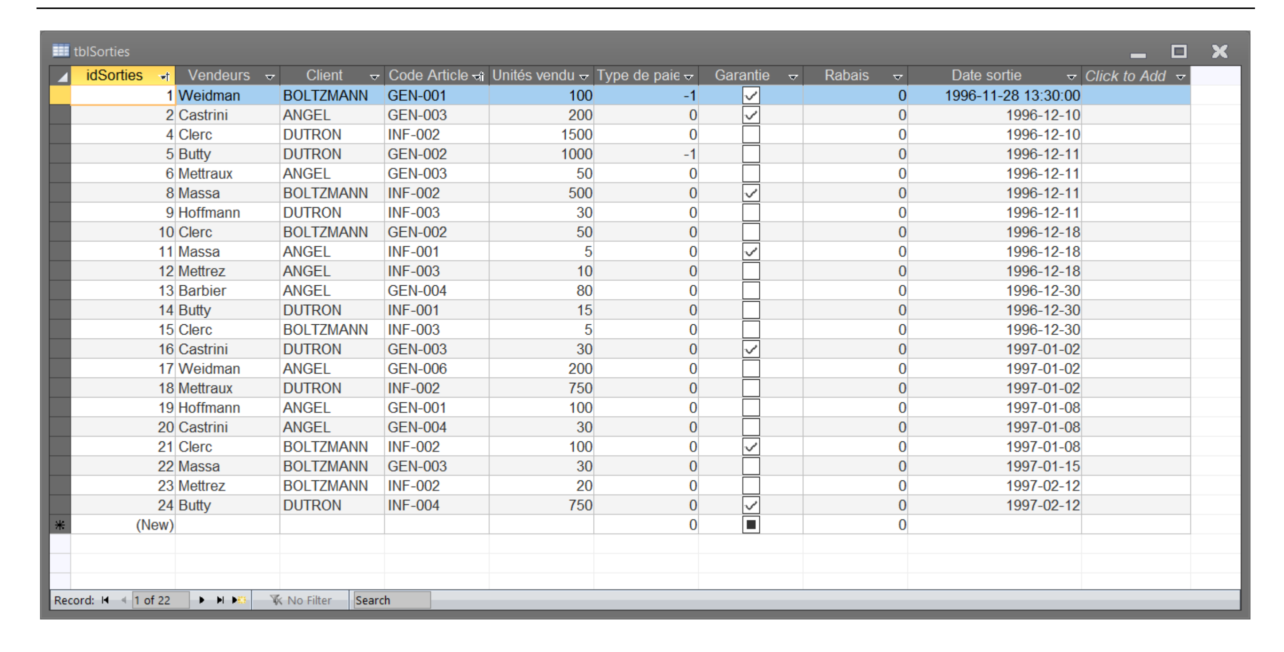
Pour ce faire, nous allons devoir faire cela avec le code SQL suivant:



qryDeleteDuplicates

```
DELETE *
FROM tblSorties
WHERE (SELECT Count(*)
    FROM tblSorties AS T
    WHERE Nz(T.idVendeurs, "") = Nz(tblSorties.idVendeurs, "")
    AND Nz(T.tblClientsId, "") = Nz(tblSorties.tblClientsId, "")
    AND Nz(T.tblArticleNb, "") = Nz(tblSorties.tblArticleNb, "")
    AND Nz(T.intUnites, "") = Nz(tblSorties.intUnites, "")
    AND Nz(T.intTypPay, "") = Nz(tblSorties.intTypPay, "")
    AND Nz(T.bolGarantie, "") = Nz(tblSorties.bolGarantie, "")
    AND Nz(T.intRabais, "") = Nz(tblSorties.intRabais, "")
    AND Nz(T.datDateOut, "") = Nz(tblSorties.datDateOut, "")
    AND T.idSorties <= tblSorties.idSorties) > 1;
```

Nous obtenons alors le résultat effectivement attendu:


tblSorties

idSorties	Vendeurs	Client	Code Article	Unités vendu	Type de paie	Garantie	Rabais	Date sortie	Click to Add
1	Weidman	BOLTZMANN	GEN-001	100	-1	☑	0	1996-11-28 13:30:00	
2	Castrini	ANGEL	GEN-003	200	0	☑	0	1996-12-10	
4	Clerc	DUTRON	INF-002	1500	0	☐	0	1996-12-10	
5	Butty	DUTRON	GEN-002	1000	-1	☐	0	1996-12-11	
6	Mettraux	ANGEL	GEN-003	50	0	☐	0	1996-12-11	
8	Massa	BOLTZMANN	INF-002	500	0	☑	0	1996-12-11	
9	Hoffmann	DUTRON	INF-003	30	0	☐	0	1996-12-11	
10	Clerc	BOLTZMANN	GEN-002	50	0	☐	0	1996-12-18	
11	Massa	ANGEL	INF-001	5	0	☑	0	1996-12-18	
12	Mettrez	ANGEL	INF-003	10	0	☐	0	1996-12-18	
13	Barbier	ANGEL	GEN-004	80	0	☐	0	1996-12-30	
14	Butty	DUTRON	INF-001	15	0	☐	0	1996-12-30	
15	Clerc	BOLTZMANN	INF-003	5	0	☐	0	1996-12-30	
16	Castrini	DUTRON	GEN-003	30	0	☑	0	1997-01-02	
17	Weidman	ANGEL	GEN-006	200	0	☐	0	1997-01-02	
18	Mettraux	DUTRON	INF-002	750	0	☐	0	1997-01-02	
19	Hoffmann	ANGEL	GEN-001	100	0	☐	0	1997-01-08	
20	Castrini	ANGEL	GEN-004	30	0	☐	0	1997-01-08	
21	Clerc	BOLTZMANN	INF-002	100	0	☑	0	1997-01-08	
22	Massa	BOLTZMANN	GEN-003	30	0	☐	0	1997-01-15	
23	Mettrez	BOLTZMANN	INF-002	20	0	☐	0	1997-02-12	
24	Butty	DUTRON	INF-004	750	0	☑	0	1997-02-12	
(New)					0	■	0		

Record: 1 of 22 No Filter Search

9.14 Requête de non correspondance

Nous allons maintenant créer une "requête de non correspondance" qui correspond en quelque sorte à la commande SQL nommée EXCEPT qui est manquant dans MS Access.

L'idée est "d'ôter" visuellement d'une sélection les lignes obtenues dans une deuxième sélection.

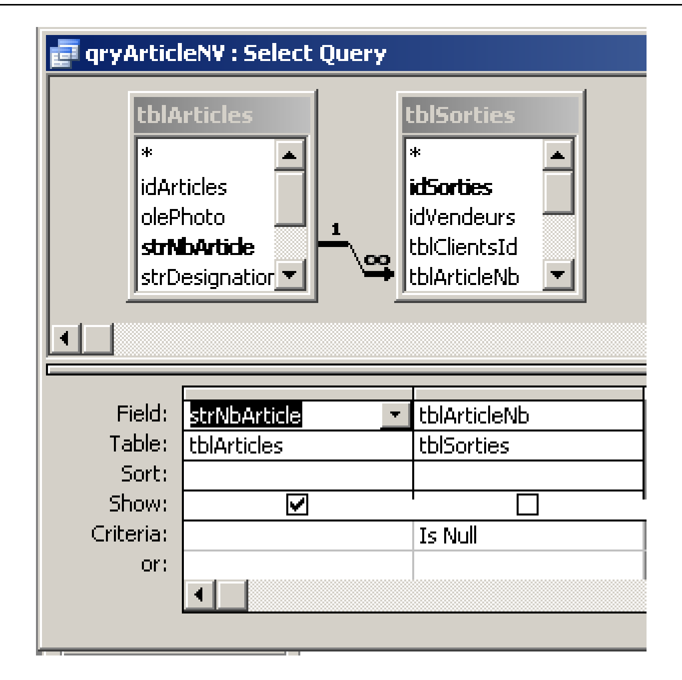
Remarque: Il s'agit aussi d'une requête très utile pour vérifier les "trous" entre deux tables qui font que l'intégrité référentielle ne puisse pas être activée. Ce type de problématique est très fréquent dans les entreprises.

Pour cela, avant tout, ajoutez dans la table *tblArticles*, un article de votre choix. Ensuite, dans la table *tblEntrees*, ajoutez une entrée de ce nouvel article.

But: Déterminer quels sont les produits qui sont en stocks mais qui n'ont jamais été vendus (non correspondance des noms d'articles *strNbArticle* entre la table *tblSorties* et *tblEntrees*).

Enregistrez la requête sous le nom *qryArticlesNV*

Résultat: vous devriez avoir comme seul non correspondant, l'article que vous venez de saisir.



Nous pouvons observer qu'il y a ici une jointure droite tout simplement !

Remarque: Pour supprimer les non-correspondants (suppression des doublons) dans le cas des problèmes d'intégrité, il suffit de supprimer le résultat montré par la requête (sélection des lignes de la requête et SUPPR).

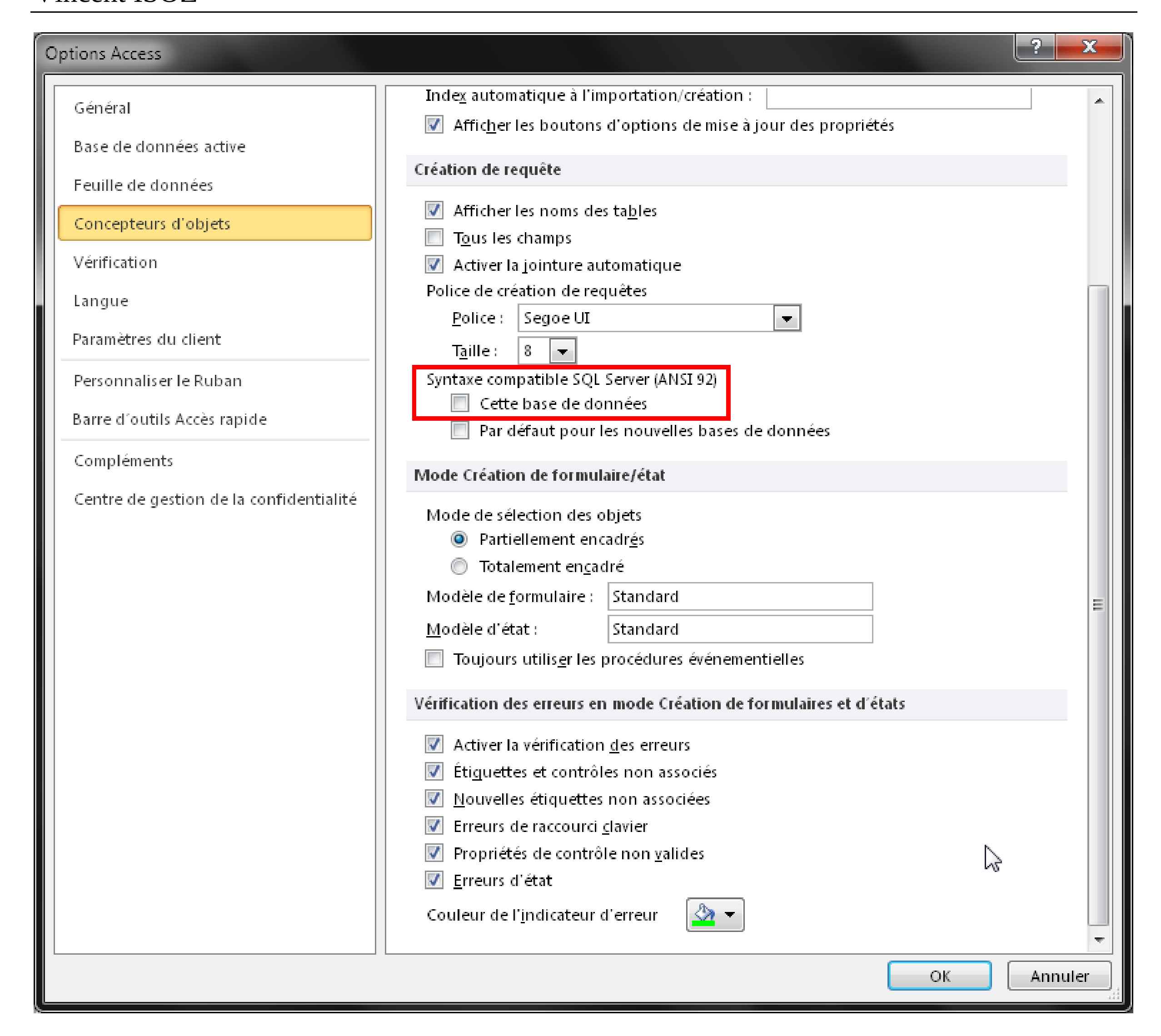
9.15 Requêtes de création/définition (mode SQL)

Veuillez maintenant créer, exécuter et comprendre les requêtes de création suivantes (elles font partie du cours "Modélisation de Base de données"):

Création d'une table simple pour les succursales (*qryCreerTable*):

```
CREATE table tblSuccursale (
idSuccursale LONG,
strVille CHAR(80),
sngPerformance CURRENCY,
CONSTRAINT idSuccursalePK PRIMARY KEY (idSuccursale) );
```

Malheureusement dans Access il faut passer par un CURRENCY à ma connaissance pour créer un champ décimal... Si on veut pouvoir utiliser du SQL standard il va falloir changer les options SQL du logiciel pour être conformes au SQL ANSI92 et alors attention aux conséquences:



Dès lors, nous pourrons par exemple:

```
ALTER TABLE MyTable
  Add COLUMN MyField DECIMAL (9,4) NULL;
```

Si nous ne voulons pas changer le moteur d'interprétation SQL il nous faudra alors passer par du VBA avec par exemple:

```
Dim conn As ADODB.Connection
Set conn = CurrentProject.Connection
conn.Execute "ALTER TABLE MyTable " _
    & "ADD COLUMN MyField DECIMAL (9,4) NULL;"
conn.Close
```

Ceci étant dit, ajoutez maintenant dans la table *tblVendeurs*, un champ nommé *tblSuccursaleId* de type Integer (*qryAltererTable*):

ALTER TABLE *tblVendeurs* ADD tblSuccursaleId LONG;

Création d'une liaison entre la table *tblSuccursale* et *tblVendeurs* (*qryCreerLiaison*):

ALTER table tblVendeurs
ADD CONSTRAINT VendeursFK FOREIGN KEY (tblSuccursaleId)
REFERENCES tblSuccursale (idSuccursale);

Ajout de données dans la table *tblSuccursale* (*qryAjoutData*):

INSERT INTO tblSuccursale (idSuccursale, strVille)
VALUES ('1','Lausanne','0.1');

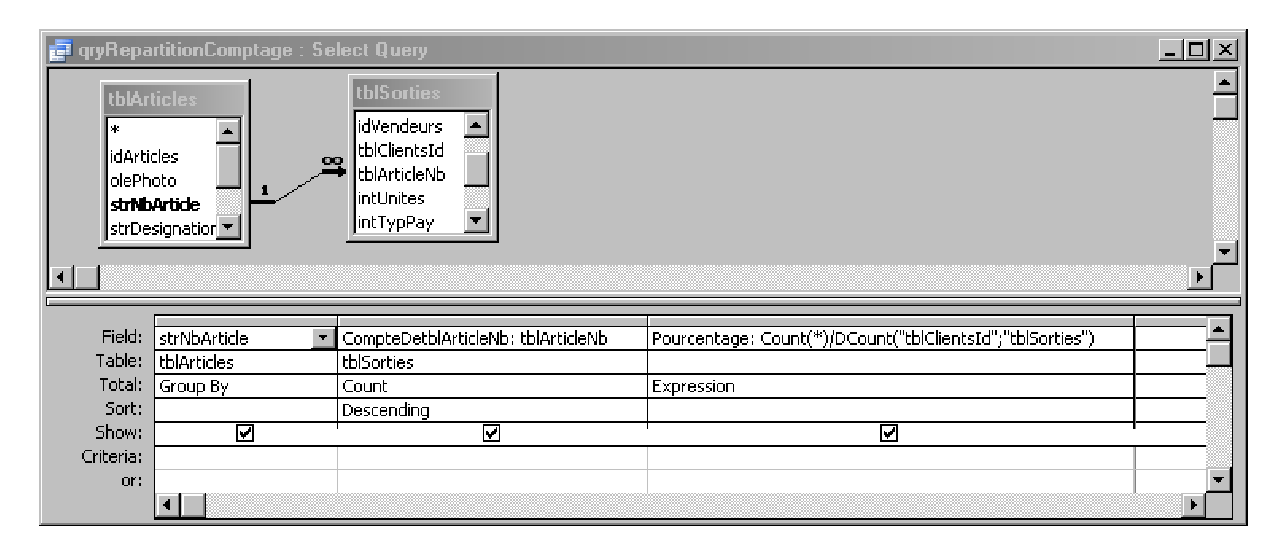
À ma connaissance il n'est pas possible (sans VBA) d'ajouter plusieurs valeurs d'un coup dans une même requête Microsoft Access... ce qui limite grandement l'exemple précédent!

Remarque: La syntaxe de cette dernière commande SQL est dans le cas général INSERT INTO nomTable (champ1, champ2, …) Values ('valeur champ1', 'valeur champ2', …)

9.16 Requête de distribution en % sur comptage

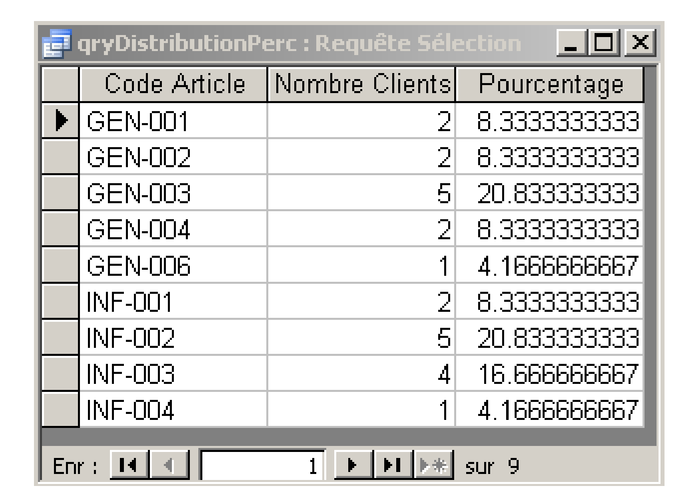
Nous continuons toujours avec notre table *tblSorties* et ce que nous désirons maintenant est connaître combien de clients ont commandé un certain produit (rien de nouveau) et savoir combien ils représentent en % du nombre total de clients ayant fait un achat.

La solution est loin d'être évidente (sans toutefois être extrêmement difficile). Nous donnons donc directement la solution:



La jointure s'expliquant par le fait que nous voulons quand même dans le listing les articles qui n'ont jamais été vendu!

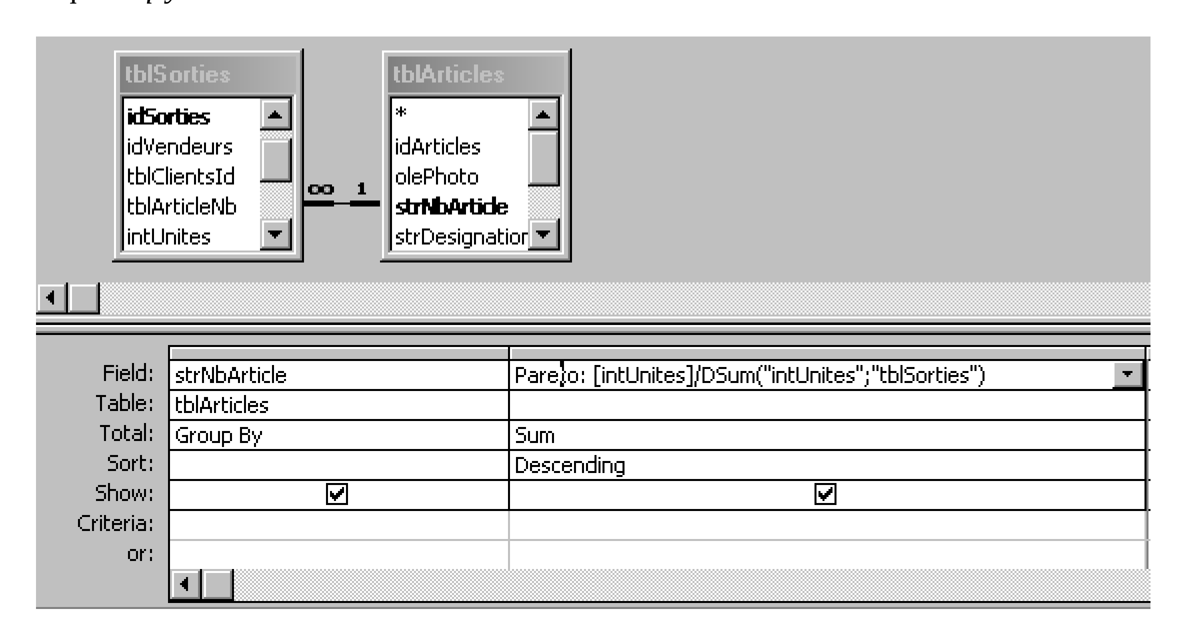
Ce qui donnera:



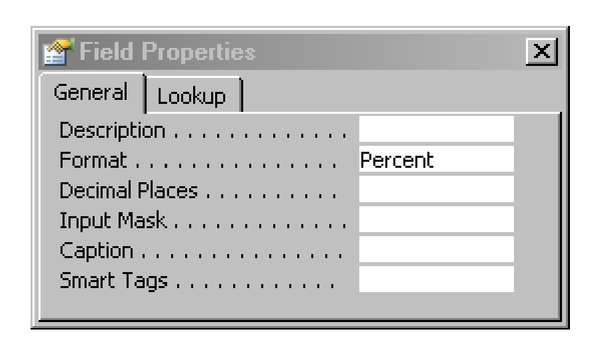
Requête que nous enregistrerons sous le nom *qryDistributionPerc*.

9.17 Requête de (Pareto) distribution en % sur somme

Nous aimerions connaître la répartition des ventes par article. Il s'agit alors de faire une requête *qryPareto* avec:



en mettant la colonne *Pareto* au format %:



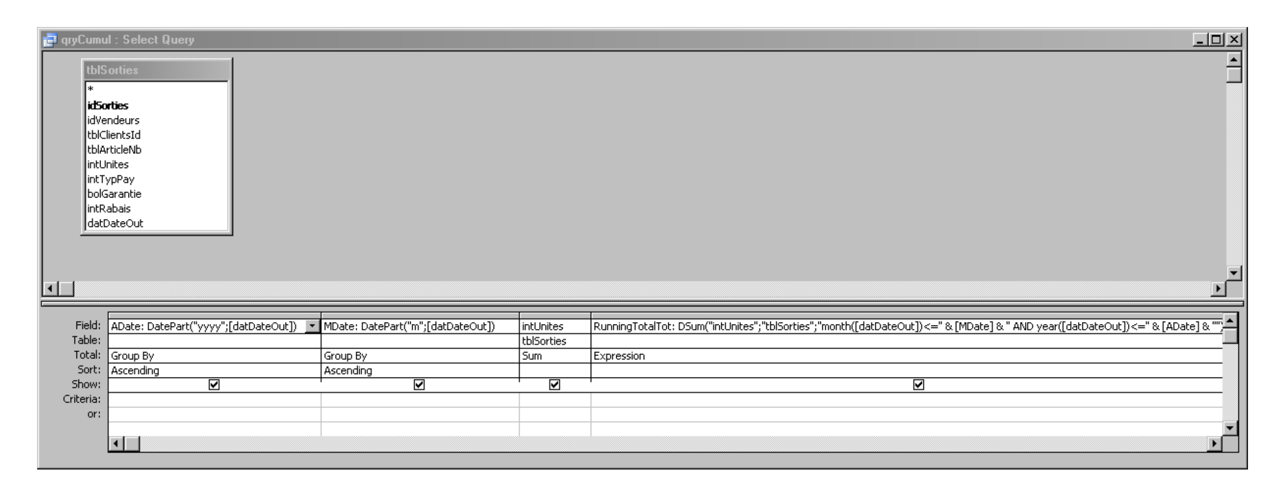
Ce qui donnera:

	Code Article	Pareto
▶	INF-002	51.30%
	GEN-002	18.77%
	INF-004	13.40%
	GEN-003	6.08%
	GEN-006	3.57%
	GEN-001	3.57%
	GEN-004	1.97%
	INF-003	0.98%
	INF-001	0.36%

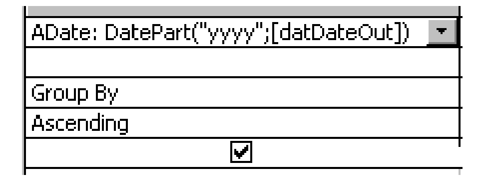
Il semblerait qu'en toute généralité, il ne soit pas possible de faire un vrai Pareto Cumulé sans du VBA dans MS Access.

9.18 Requête de cumul chronologique

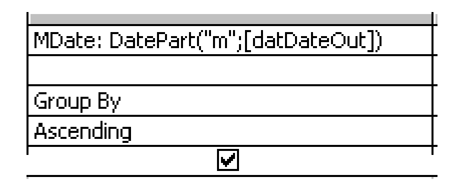
L'idée est de créer une requête qui cumula la vente des unités mensuelles avec un recommencement chaque année. Pour ce faire, la seule possibilité simple est la suivante:



où nous avons dans la première colonne:



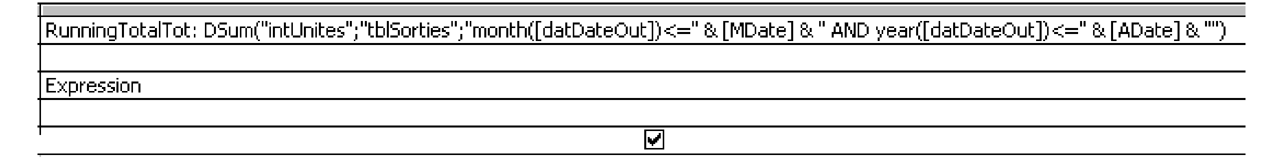
Dans la deuxième:



Dans la troisième:



Enfin dans la dernière:



Attention!!! MS Access ne gère pas bien les fonctions en français. Il faut donc écrire tout en anglais afin que la requête ci-dessous fonctionne correctement!

9.19 Requête système

Depuis MS Access 2007, un requête SQL que j'affectionne particulièrement est la suivante:

```
SELECT MSysObjects.Name,IIF([Type]=4,'Linked','Non-Linked') As Status,
DCount("*",[MSysObjects].[Name]) AS[Record Count]
FROM MSysObjects
WHERE (((Left([Name],1))<>"~") AND ((Left([Name],4))<>"MSys") AND
((MSysObjects.Type) In (1,4,6)))
ORDER BY MSysObjects.Name;
```

Cette requête listera toutes les tables avec le nombre de lignes qu'elles contiennent. C'est infiniment utile soit à des fins de statistiques, soit pour créer un identifiant global (qui manque cruellement à MS Access excepté sous forme de GUID).

Si jamais, voici déjà l'équivalent en VBA (il m'a semble plus judicieux de déjà l'indiquer ici plutôt que dans la section consacrée au VBA):

```
Sub CountAll()
    Dim tdf As DAO.TableDef

    For Each tdf In CurrentDb.TableDefs
      If Len(tdf.Connect) > 0 Then    'it is a Linked Table (this is green out)
        Debug.Print tdf.Name & "[Linked] has " & DCount("*", tdf.Name) & " Records!"
      Else
        Debug.Print tdf.Name & "[Non-Linked] has " & DCount("*", tdf.Name) & " Records!"
      End If
    Next

End Sub
```

9.20 Requêtes de statistiques non natives

Contrairement à Oracle SQL le SQL de Microsoft Access est très pauvre en requêtes statistiques (comme son grand frère SQL Server par ailleurs).

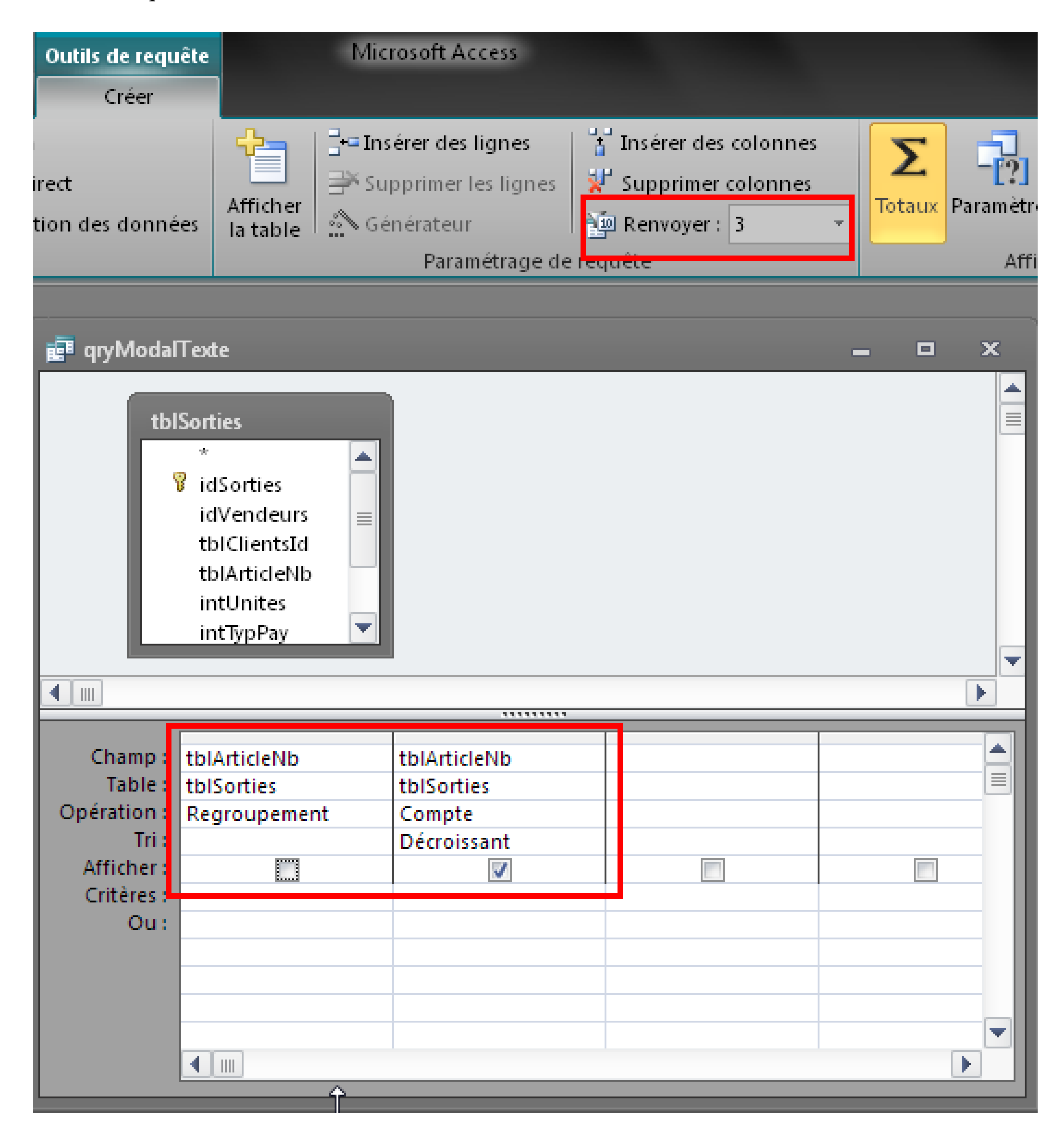
Nous allons voir ici comment faire certaines statistiques courantes en utilisant si aussi peu que possible ou pas du tout de VBA.

Les requêtes sont présentées dans l'ordre croissant de difficulté.

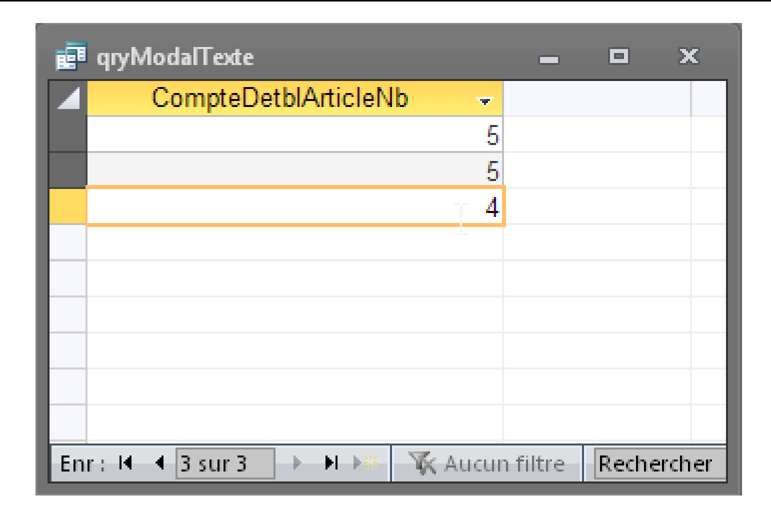
9.20.1 Requête de valeur modale

Une statistique élémentaire (outre la moyenne, l'écart-type et l'étendue) régulièrement demandée dans les requêtes est l'obtention des valeurs modales.

Voyons comment obtenir dans un premier temps les trois premières valeurs modales d'une valeur numérique:



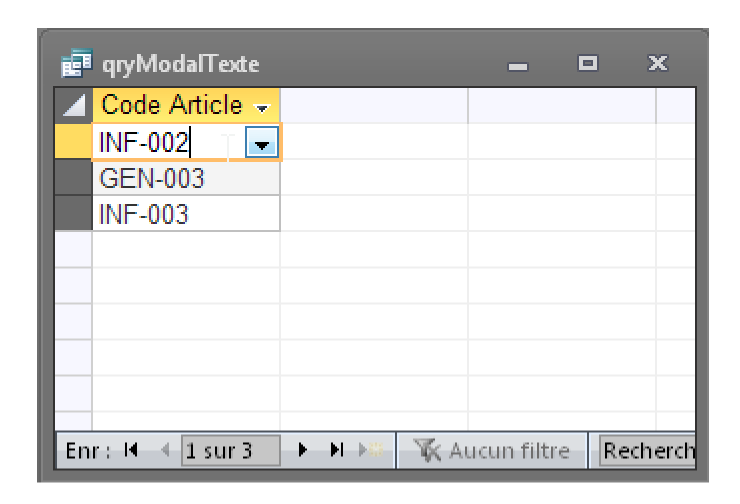
Ce qui donne:



Et pour les articles modaux (donc ceux apparaissent le plus de fois):

Champ :	tblArticleNb	tblArticleNb	
Table :	tblSorties	tblSorties	
Opération :	Regroupement	Compte	
Tri :		Décroissant	
Afficher :	☑	☐	☐
Critères :			
Ou :			

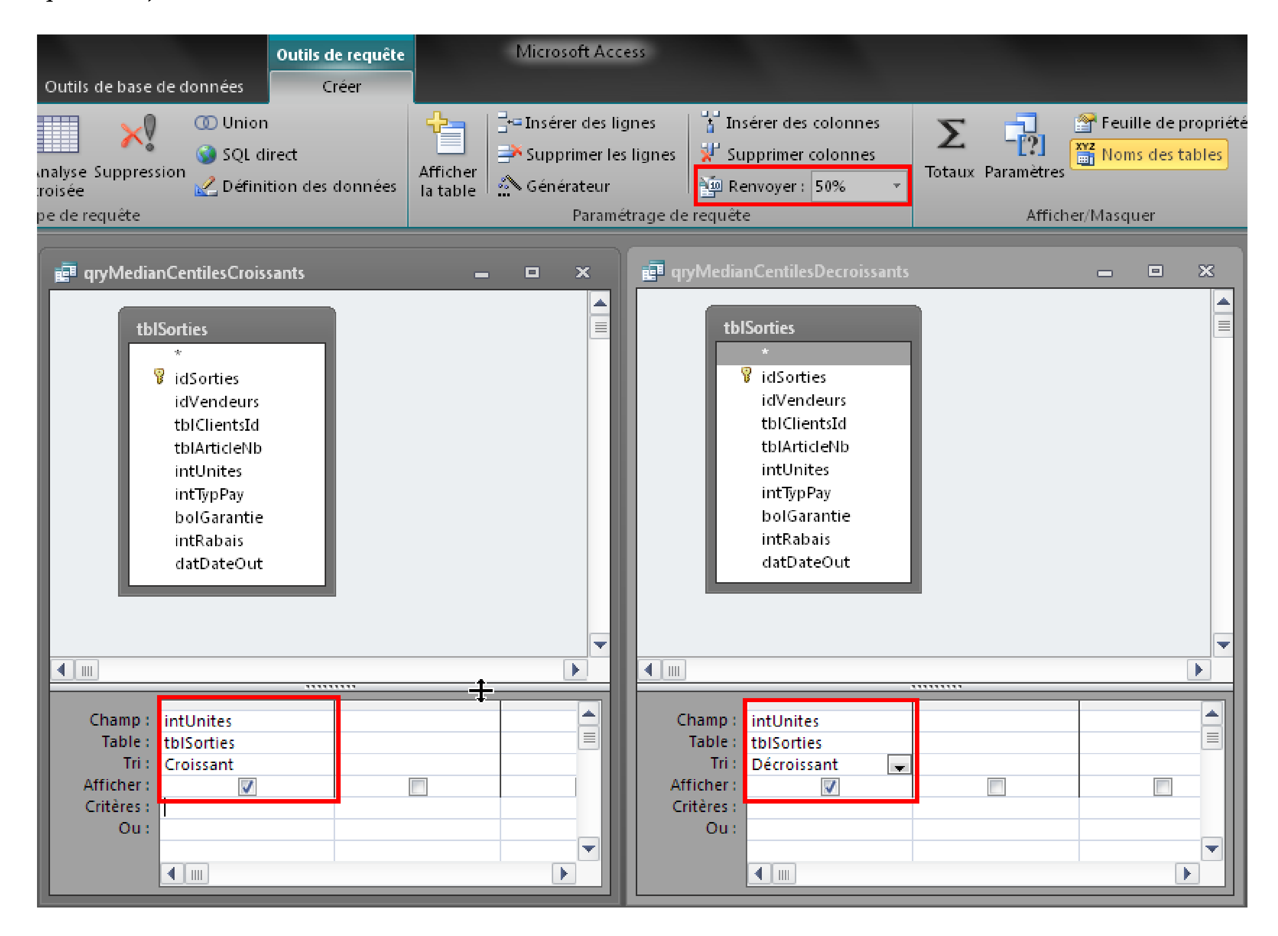
Ce qui donne:



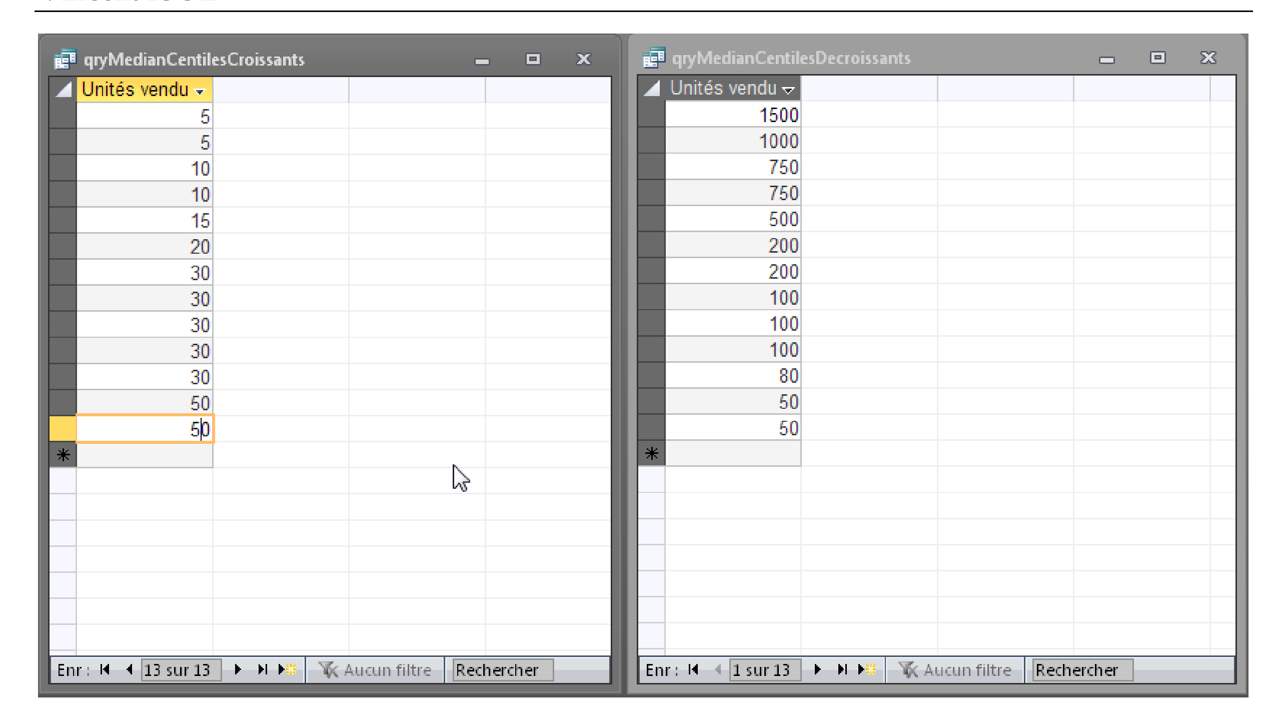
9.20.2 Requête d'obtention de la médiane

C'est une analyse souvent demandée. Il existe de nombreuses solutions en VBA mais voici une solution utilisant que des requêtes Access mais au nombre de 5... (bon après il est possible de la réduire à 1 en imbriquant les requêtes bien évidemment!):

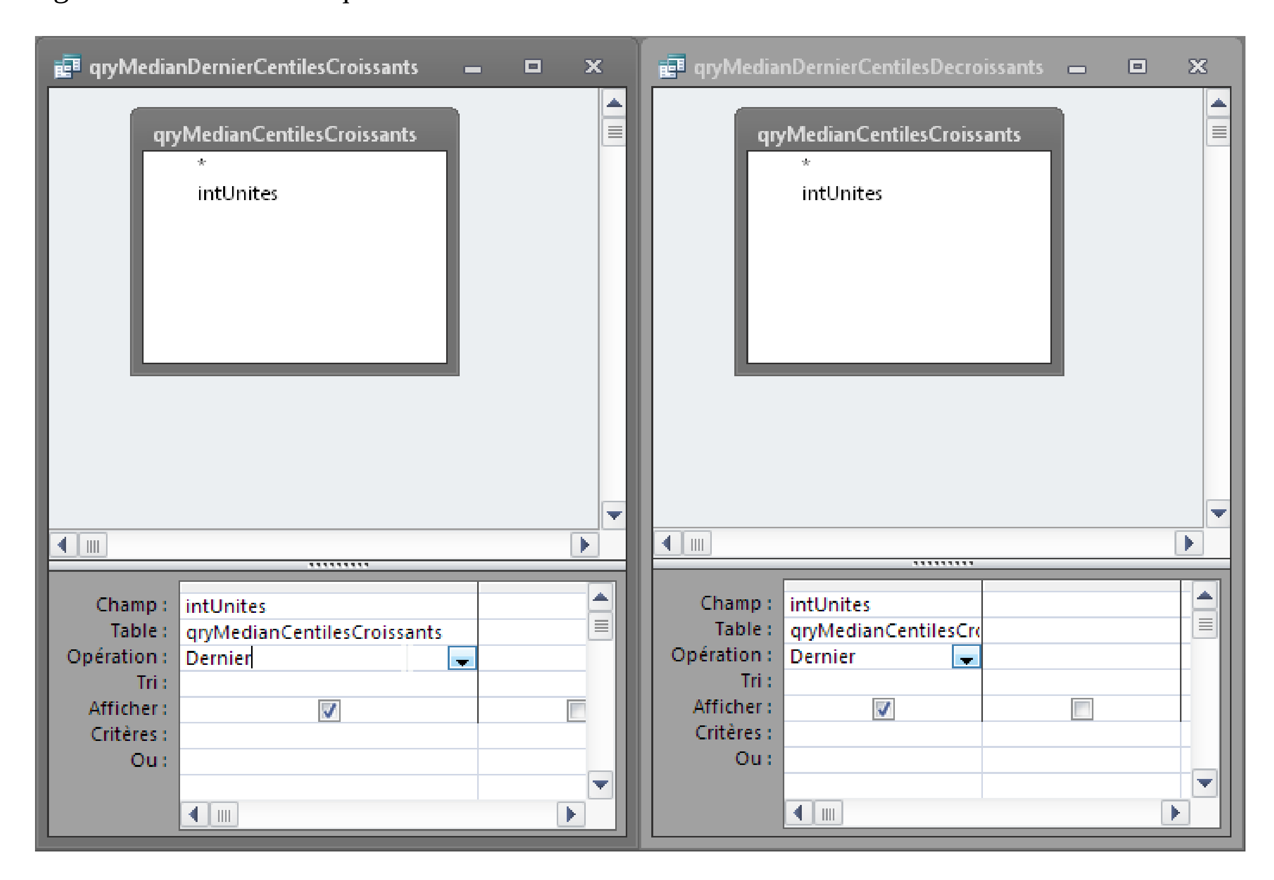
L'astuce consiste d'abord à faire deux requêtes qui respectivement extraient le premier 50 centile (les deux premiers quartiles) et le deuxième 50 centile (les troisième et quatrième quartiles).



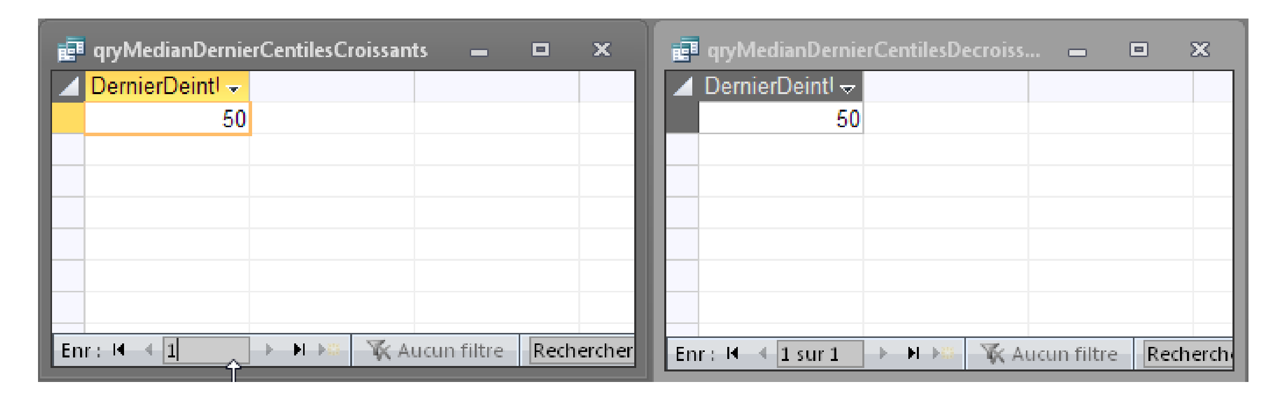
Ce qui donne respectivement:



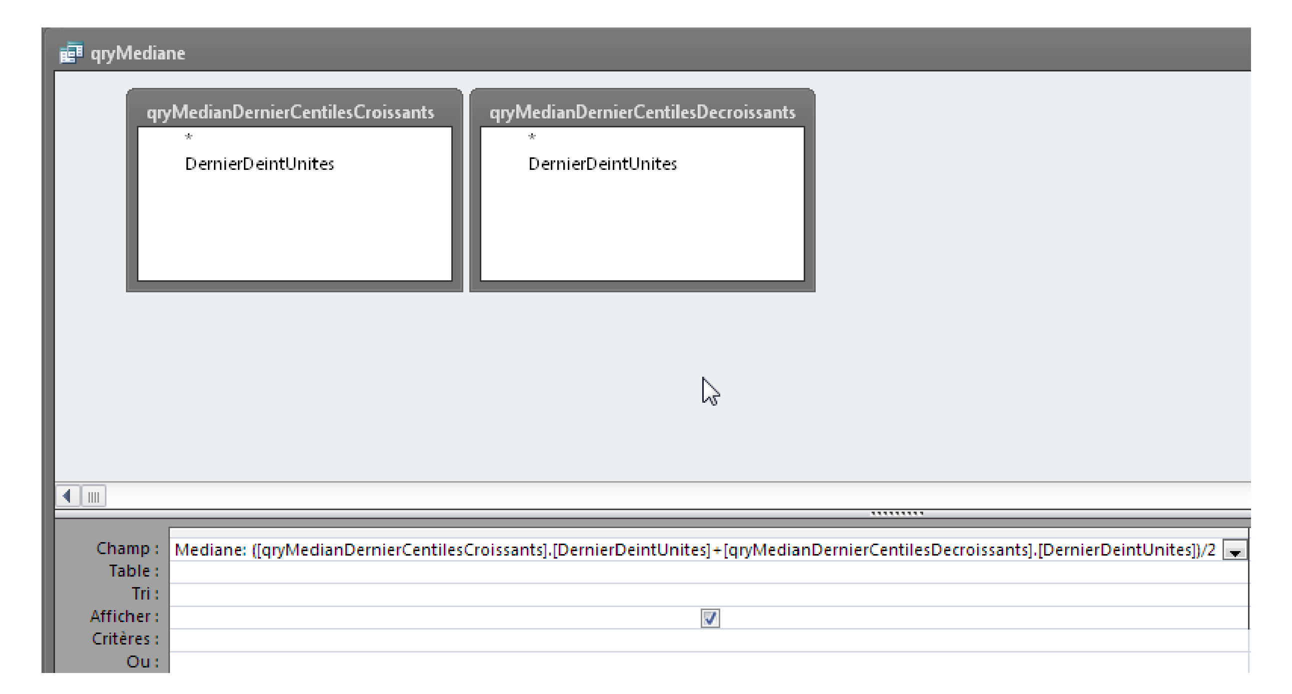
Maintenant il nous faut récupérer la dernière ligne de la première requête et aussi la dernière ligne de la deuxième requête.



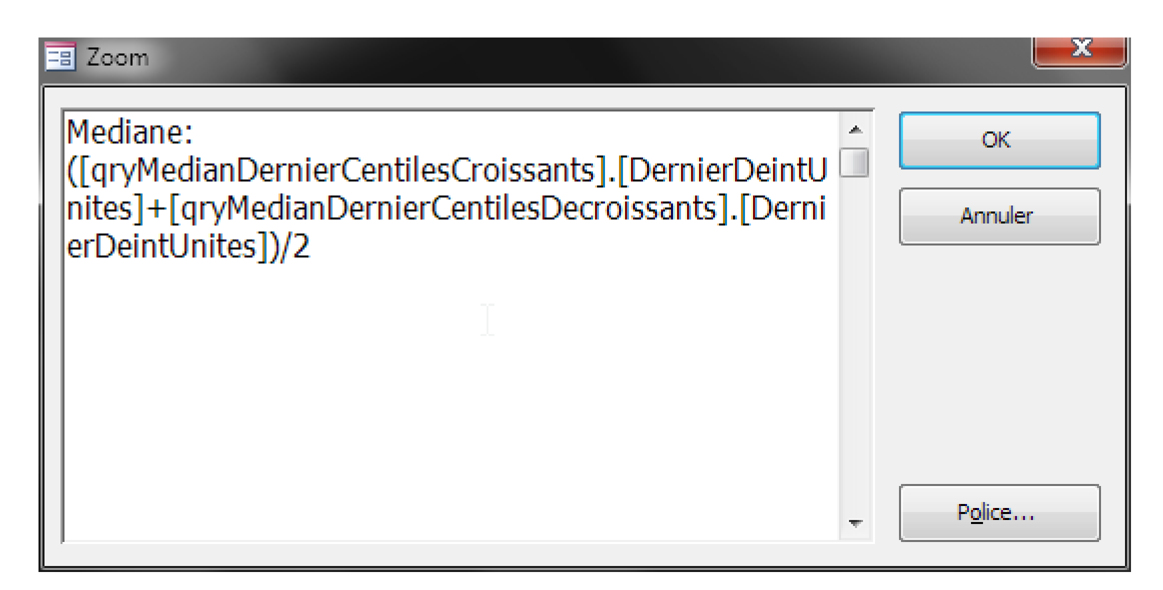
Ce qui donne:



Maintenant passons à la dernière requête:



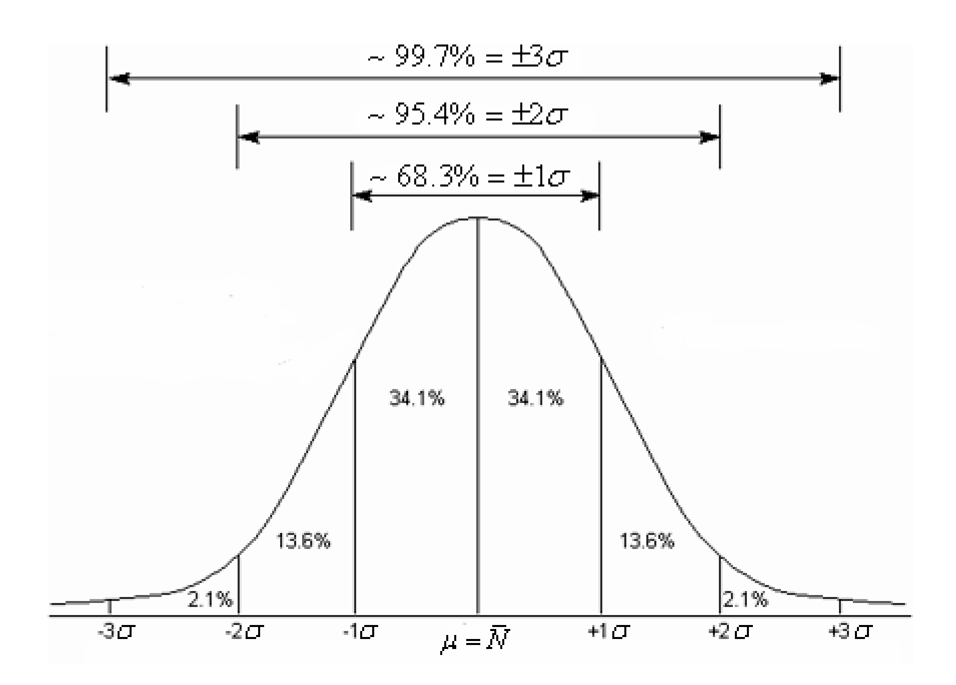
Soit en détails:



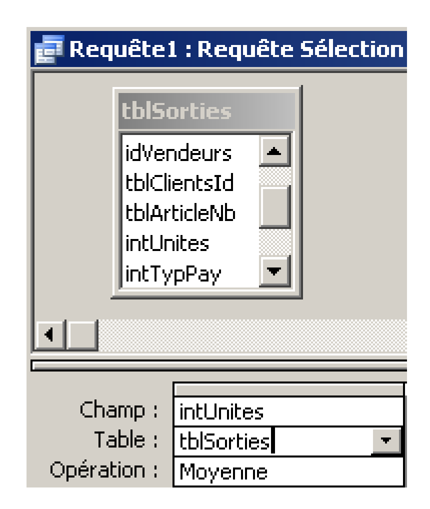
Ce qui donne à l'exécution.

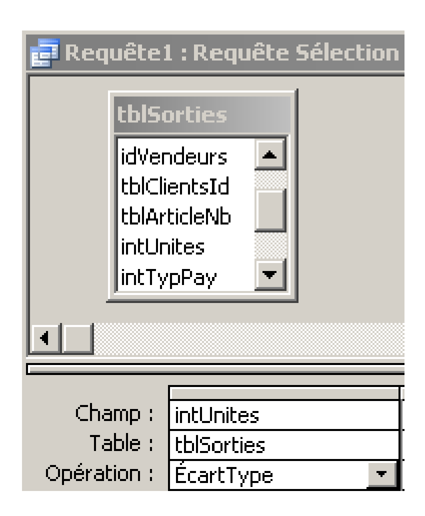
9.20.3 Requêtes d'intervalle de fluctuation

On peut faire de la statistique inférentielle assez facilement avec MS Access en jouant avec les sous-requêtes. Pour cela, imaginons que nous souhaitons voir tous les articles dont les unités vendues supposées suivre une distribution de type gaussienne ont une moyenne connue (à l'aide d'une requête de regroupement) et qui se situent au-delà de $\geq \mu + 3\sigma$.



La première étape consiste à créer les simples requêtes suivantes et à copier leur code SQL respectif:

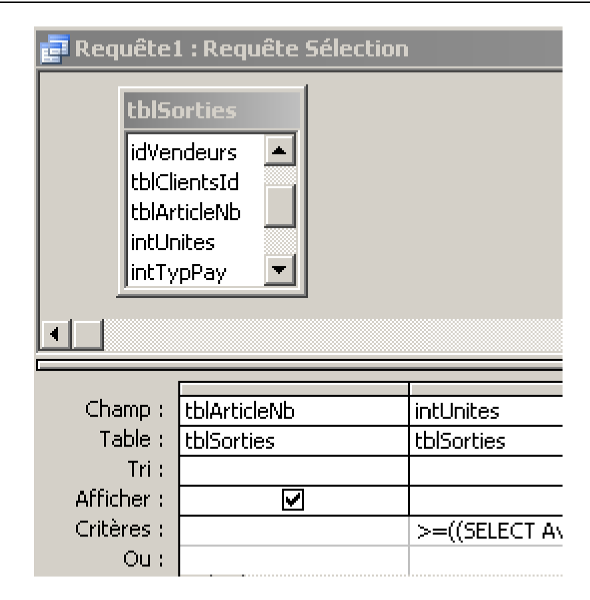




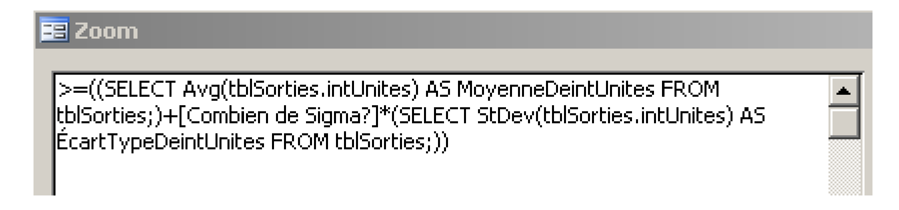
SELECT Avg(tblSorties.intUnites) AS MoyenneDeintUnites FROM tblSorties;

SELECT StDev(tblSorties.intUnites) AS ÉcartTypeDeintUnites FROM tblSorties;

Ensuite, nous faisons une sous requête simple du type:



Dont le critère détaillé est (zoom):

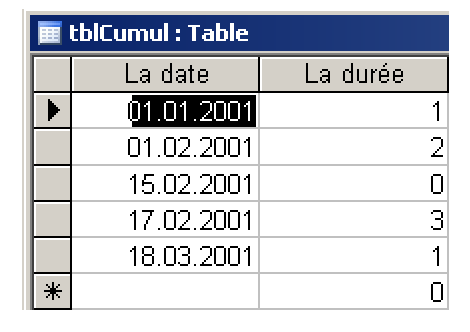


Enregistrez cette requête sous le nom *qryStatistiques*

9.20.4 Requêtes moyenne mobile et somme cumulée

Nous allons maintenant sortir un peu de notre base de données classique (bien que l'exemple qui va venir puisse quand même s'y appliquer…).

Considérons la table suivante:


tblCumul : Table

La date	La durée
01.01.2001	1
01.02.2001	2
15.02.2001	0
17.02.2001	3
18.03.2001	1
	0

Supposons que nous aimerions la valeur cumulée et la moyenne mobile par ligne d'enregistrement afin d'obtenir:

	La date	La durée	Cumul	Moyenne Mobil
▶	01.01.2001	1	1	1
	01.02.2001	2	3	1.5
	15.02.2001	0	3	1
	17.02.2001	3	6	1.5
	18.03.2001	1	7	1.4
*		0		

Pour cela (c'est scandaleux à mon avis mais c'est ainsi…) il vous faudra d'abord créer un module VBA (peu importe son nom) avec le code ci-dessous:

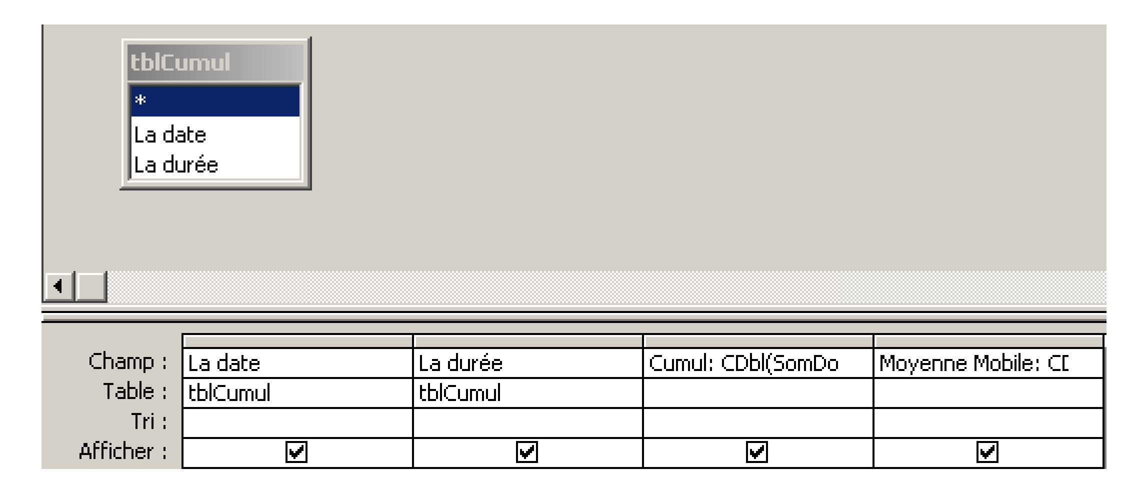
```
Option Compare Database

Function DateUS(ByVal dt As Variant)

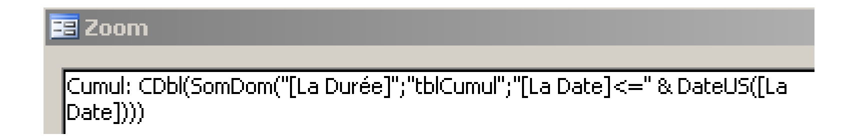
    If IsNull(dt) Then Exit Function
    DateUS = "#" & Month(dt) & "/" & Day(dt) & "/" & Year(dt) & "#"

End Function
```

Une fois ceci fait, voici la requête à créer:



avec les détails de la fonction *Cumul*:



et les détails de la fonction *Moyenne Mobile*:

Zoom

Moyenne Mobile: CDbl(SomDom("[La Durée]";"tblCumul";"[La Date]<=" & DateUS([La Date]))/CpteDom("[La Durée]";"tblCumul";"[La Date]<=" & DateUS([La Date])))

9.20.5 Requêtes de data mining/machine learning

Dans le Data Mining certaines méthodes (comme les réseaux de neurones) se basent sur le principe que l'on a une série de données pour entraîner les neurones et déterminer les meilleurs perceptrons qui permettront par la suite de faire de l'analyse.

Souvent ces données sont soit choisies manuellement, soit au hasard. Voyons alors comment dans notre table *tblSorties* tirer 5 sorties au hasard.

qrySelectAlea : Requête Sélection

```
SELECT TOP 5 tblSorties.* FROM tblSorties ORDER BY Rnd([idSorties]);
```

Une fois que nous avons les données d'entraînement, nous devons tester le modèle théorique sur les données restantes. Voyons comment maintenant ne sélectionner que les données restantes:

ToDo: à faire...

9.21 Analyse simple de portefeuilles

Pour cet exemple, précieux aux banquiers débutants, considérons que notre petit magasin joue sur le marché des actifs financiers et possède trois titres en proportions égales (pour simplifier…), soit 1/3, dans un portefeuille et *n* observations de leur rendement $R_{i,j}$.

Donc soit la table *tblPortfolio* suivante:

tblPortfolio : Table

idObs	sngTitre1	sngTitre2	sngTitre3
1	-0.15	0.29	0.38
2	0.05	0.18	0.63
3	-0.43	0.24	0.46
4	0.79	0.25	0.36
5	0.32	0.17	-0.57

Première tâche que nous allons effectuer: Afficher dans une petite boîte de dialogue l'espérance de chaque titre *i* selon la relation triviale:

$$E(R_i) = \hat{\mu}_i = \frac{\sum_{j=1}^{n} R_{i,j}}{n}$$

et l'écart-type au carré (variance) de chaque titre *i* selon la relation tout aussi triviale:

$$\hat{\sigma}_i^2 = \frac{\sum_{j=1}^{n} \left(R_{i,j} - \hat{\mu}_i\right)^2}{n-1}$$

et le rendement moyen par:

$$E\left(R_p\right)=\sum_{i=1}^{n} X_i R_i = X_1\hat{\mu}_1 + X_2\hat{\mu}_1 + X_3\hat{\mu}_1$$

et l'écart-type du portefeuille au carré du portefeuille (variance) par la relation un peu moins triviale:

$$V\left(R_p\right)=\sigma^2\left(R_p\right)=\sum_{i=1}^{n} X_i^2 V\left(R_i\right)+2\sum_{i\neq}\sum_{j} X_i X_j \operatorname{cov}\left(R_j,R_i\right)$$
$$=\sum_{i=1}^{n} X_i^2\sigma_i^2+2\sum_{i\neq}\sum_{j} X_i X_j \operatorname{cov}\left(R_j,R_i\right)$$

Ceci est fort amusant à programmer et donne:

```
Option Compare Database
Option Explicit

Sub Portfolio()

    Dim dbs As Database
    Dim rstTit As DAO.Recordset
    Dim i, j, k, l, intTitle, nb As Integer
    Dim strTitle, strTitle1, strTitle2, strMessage As String
    Dim sngRendMoyen, sngVarFstTerm, sngVarSndTerm, sngProdEsp As Single
    Dim ArrayPort()

    intTitle = 3
    Set dbs = CurrentDb
    Set rstTit = dbs.OpenRecordset("tblPortfolio", dbOpenTable)
    ReDim ArrayPort(2, intTitle)

    nb = DCount("*", "tblPortfolio")
    MsgBox nb & " obervations des trois titres", vbOKOnly, "TSA Portfolio"

    'Nous calculons l'espérance du rendement de chaque titre
    For i = 1 To intTitle
        j = 0
        strTitle = "sngTitre" & i
        With rstTit
             rstTit.MoveFirst
             Do Until .EOF
                 j = j + 1
                 ArrayPort(1, i) = ArrayPort(1, i) + rstTit(strTitle)
                 .MoveNext
```

```
            Loop
                ArrayPort(1, i) = ArrayPort(1, i) / j
                Debug.Print ArrayPort(1, i)
        End With
    Next i

    'On affiche le résultat des calculs de l'esperance dans une boîte simple
    For i = 1 To intTitle
        strMessage = strMessage & "Esperance de rendement du titre " & i & " : " & ArrayPort(1, i) & vbCrLf & vbCrLf
    Next i
    MsgBox strMessage, vbOKOnly + vbInformation

    'Nous calculons l'écart-type du rendement de chaque titre
    For i = 1 To intTitle
        j = 0
        strTitle = "sngTitre" & i
        With rstTit
            rstTit.MoveFirst
            Do Until .EOF
                j = j + 1
                ArrayPort(2, i) = ArrayPort(2, i) + (rstTit(strTitle) - ArrayPort(1, i)) ^ 2
                .MoveNext
            Loop
                ArrayPort(2, i) = ArrayPort(2, i) / (j - 1)
                Debug.Print ArrayPort(2, i)
        End With
    Next i

    'On affiche le résultat des calculs de l'écart-type ET l'espérance dans une boîte simple
    For i = 1 To intTitle
        strMessage = strMessage & "Variance de rendement du titre " & i & " : " & ArrayPort(2, i) & vbCrLf & vbCrLf
    Next i
    MsgBox strMessage, vbOKOnly + vbInformation

    'Nous calculons le rendement moyen
    For i = 1 To intTitle
        sngRendMoyen = 1 / intTitle * ArrayPort(1, i) + sngRendMoyen
    Next i
    strMessage = strMessage & "Rendement espéré du portefeuille : " & sngRendMoyen & vbCrLf & vbCrLf
    MsgBox strMessage

    'Nous calculons le premier terme de la variance globale du portefeuille qui est très simple
    For i = 1 To intTitle
        sngVarFstTerm = sngVarFstTerm + ArrayPort(2, i) * (1 / intTitle) ^ 2
    Next i

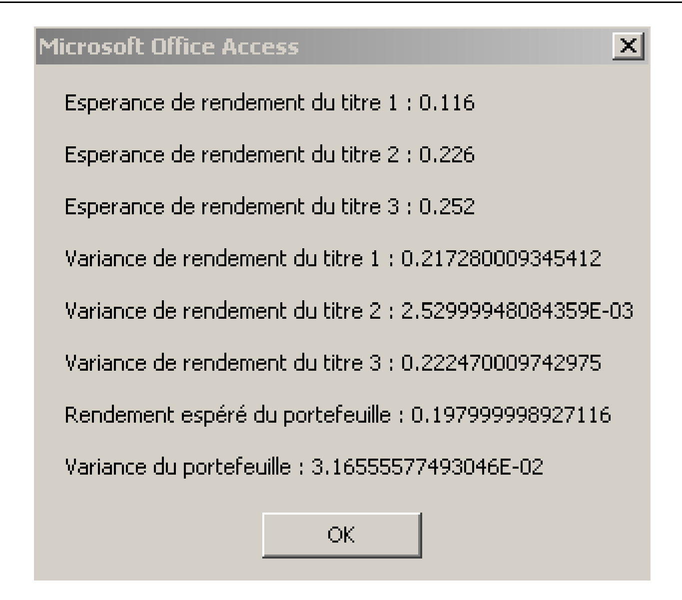
    'Nous calculons le deuxième terme intégrant la covariance et en utlisant sa propriété de linéarité
    For i = 1 To intTitle
        For j = i To intTitle
            If j <> i Then
                k = 0
                'Première chose nous calculons l'espérance du produit
                strTitle1 = "sngTitre" & i
                strTitle2 = "sngTitre" & j
                With rstTit
                    rstTit.MoveFirst
                    Do Until .EOF
                        sngProdEsp = rstTit(strTitle1) * rstTit(strTitle2)
                        .MoveNext
                        k = k + 1

                    Loop
                End With
                sngProdEsp = sngProdEsp / k
                sngVarSndTerm = sngVarSndTerm + (1 / intTitle) ^ 2 * (sngProdEsp - ArrayPort(1, j) * ArrayPort(1, i))
            End If
        Next j
    Next i

    strMessage = strMessage & "Variance du portefeuille : " & (sngVarSndTerm + sngVarFstTerm)
    MsgBox strMessage

End Sub
```

et le résultat donne pour la dernière boîte de message:



9.22 Transactions

Les systèmes de bases de données professionnels comme Oracle, SQL Server, mySQL etc. ont un concept que l'on appelle les "transactions" (voir mon e-book sur le SQL pour plus de détails). L'idée est simple: si la requête n'est pas parfaitement exécutée, tous les changements effectués par celle-ci sont annulés et l'utilisateur en est averti (dans de nombreux cas pratique un tel contrôle est primordial pour ne pas se retrouver dans certaines situations suicidaires...). C'est donc quelque chose d'extrêmement important lorsque l'on manipule des données sensibles et que l'on veut garantir l'intégrité de la base de données dans son ensemble.

Le problème (parmi d'autres...) avec MS Access c'est que les transactions ne peuvent être faites à ce jour qu'en VBA... c'est aussi une des raisons (parmi de nombreuses autres) pour lesquelles certains développeurs de métier créent tout en VBA dans MS Access.

Donc voici un exemple complet par rapport à notre magasin:

```
Sub ListReferences()

Public Sub TransferFunds()
    Dim wrk As DAO.Workspace
    Dim db As DAO.Database

    Set wrk = DBEngine(0)
    Set db = CurrentDb

    On Error GoTo trans_Err

    wrk.BeginTrans

    'Attention les transactions respectent les clés/primaires et étrangères
ainsi que les champs requis mais pas les masques!
    db.Execute "INSERT INTO
tblArticles(strNbArticle,strDesignation,tblFournisseurNom,intMoq,intUnitPri
ce) VALUES ('GEN-010','<div>Test insertion 1</div>','1',25,30)",
dbFailOnError
    db.Execute "INSERT INTO
tblArticles(strNbArticle,strDesignation,tblFournisseurNom,intMoq,intUnitPri
ce) VALUES ('GEN-011','<div>Test insertion 2</div>','1',25,30)",
dbFailOnError
```

```
    'ici on provoque l'erreur
    db.Execute "INSERT INTO
tblArticles(strNbArticle,strDesignation,tblFournisseurNom,intMoq,intUnitPri
ce) VALUES ('','<div>Test insertion 3</div>','1',25,30)", dbFailOnError
    'on reprend avec une requête normale
    db.Execute "INSERT INTO
tblArticles(strNbArticle,strDesignation,tblFournisseurNom,intMoq,intUnitPri
ce) VALUES ('GEN-013','<div>Test insertion 4</div>','1',25,30)",
dbFailOnError

    wrk.CommitTrans dbForceOSFlush

trans_Exit:
    wrk.Close
    Set db = Nothing
    Set wrk = Nothing
    Exit Sub

trans_Err:
    wrk.Rollback
     If DBEngine.Errors.Count > 0 Then
       For Each errLoop In DBEngine.Errors
          MsgBox "Error number: " & errLoop.Number & vbCr & _
             errLoop.Description
       Next errLoop
    End If
    MsgBox "Opération annulée. Il y a eu une erreur parmi les requêtes
exécutées", vbCritical + vbOKOnly
    Resume trans_Exit

End Sub
```

Ainsi, si une des requêtes, et une seule seulement (!) venait à échouer, l'ensemble est annulé! Ce qui est fort heureux mais qui devrait être intégré par défaut dans le logiciel.

10 Fonctions

Nous avons rencontré jusqu'à maintenant plusieurs fonctions de calcul bien utiles dans les requêtes, formulaires et dans les tables (pour les valeurs par défaut).

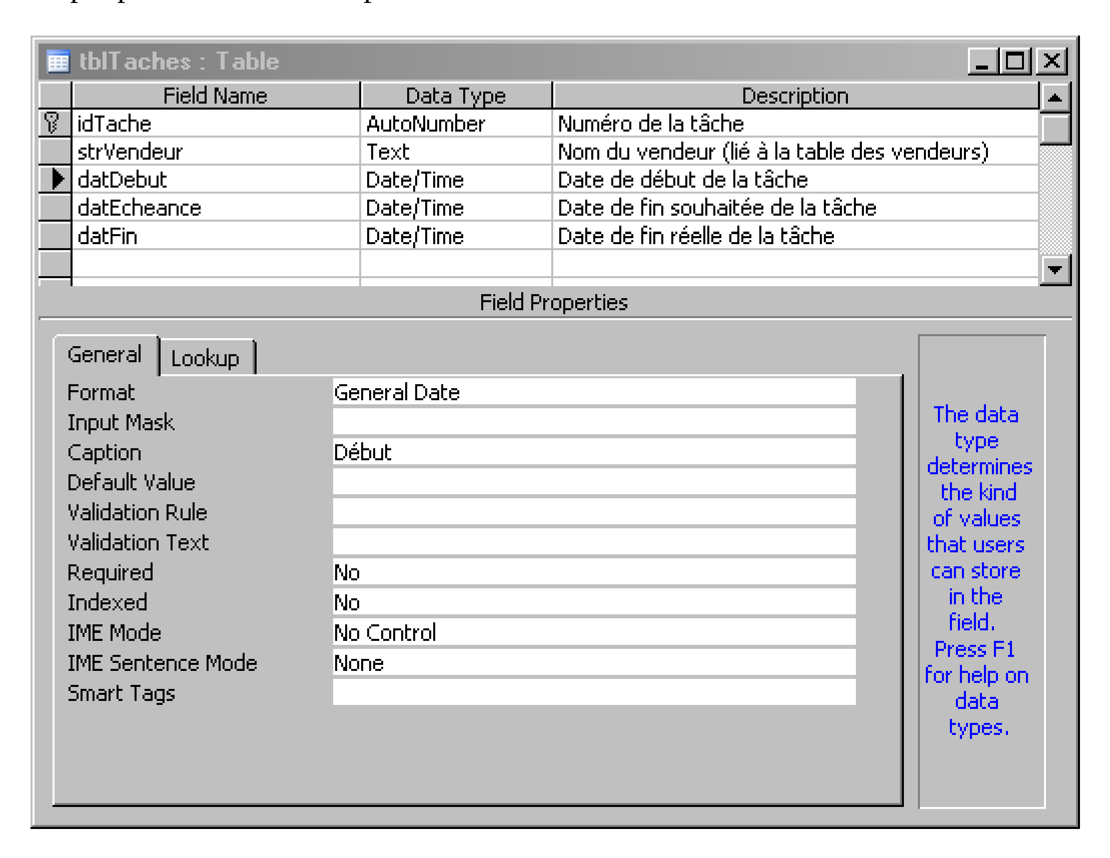
Le but de ce chapitre, étant donné l'importance relative du sujet, est de recompiler les fonctions déjà vue et d'en montrer de nouvelles qui sont aussi utilisées dans les entreprises.

10.1 Relation d'ordre comme validation d'un record (table)

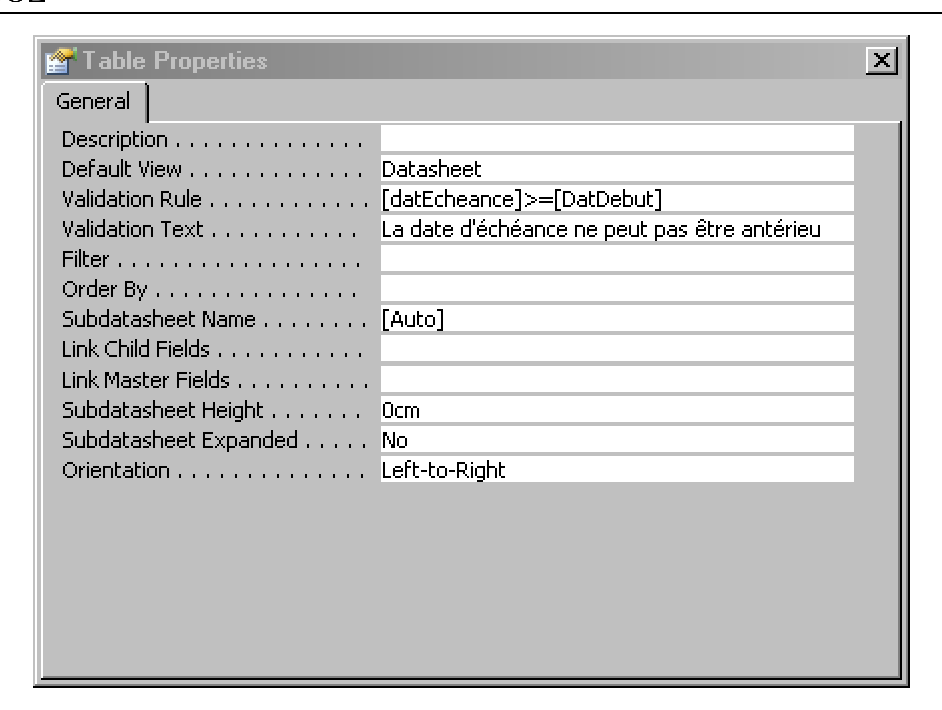
Dans l'option de validation de champ d'une table il n'est pas possible de faire référence à un autre champ de la même table (ou même d'une autre table).

Cependant dans des cas simples il est possible d'aller dans les propriétés de la table pour définir une règle de validation globale.

Par exemple pour une table comprenant des tâches comme ci-dessous:

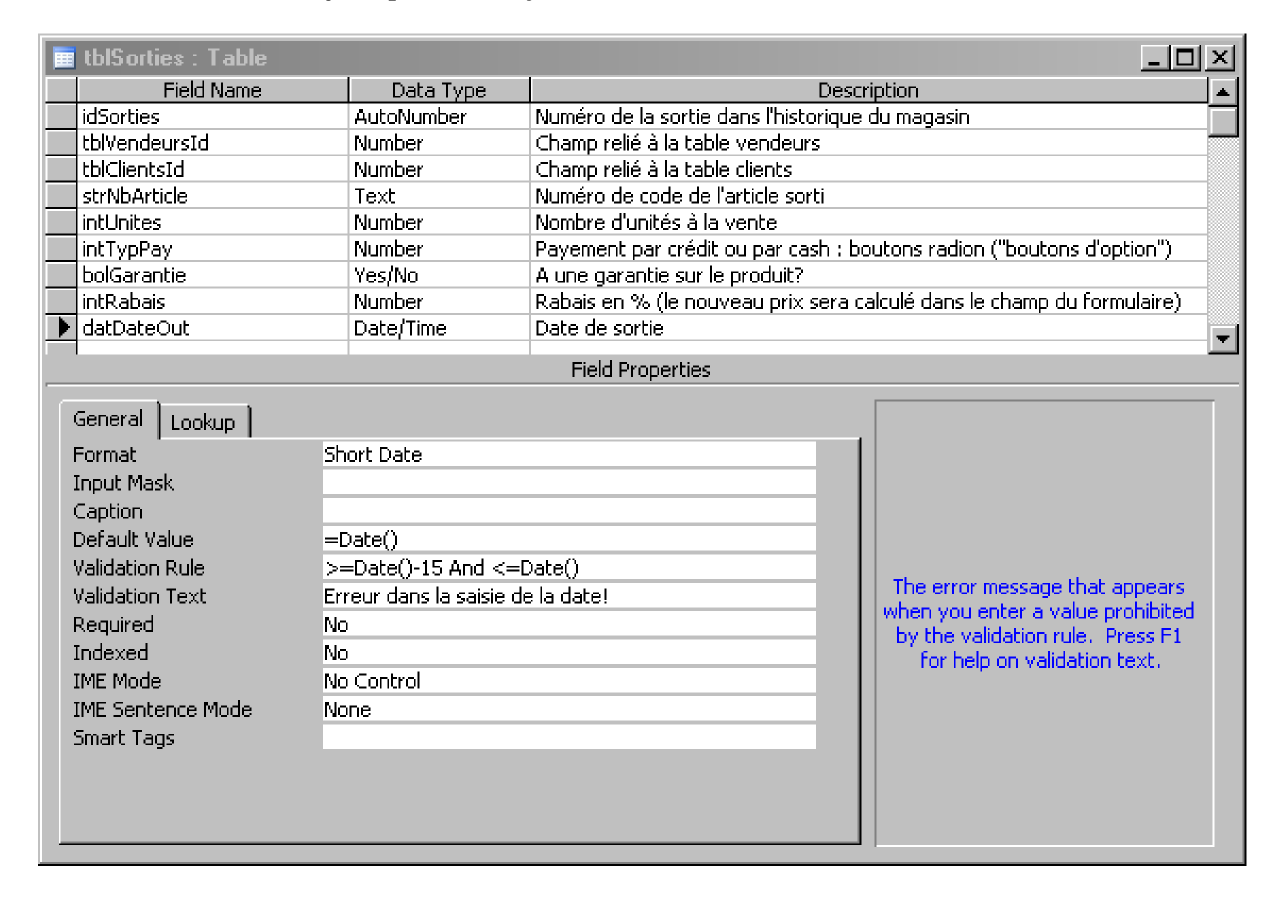


nous souhaiterions que la date d'échéance ne puisse pas être antérieur à la date de début. Pour cela nous allons cliquer sur le bouton dans la barre d'outils pour faire apparaître la boîte de dialogue suivante:



10.2 Fonctions Date comme valeur par défaut (tables)

Dans la table *tblSorties* nous souhaiterions qu'à chaque nouvelle saisie soit automatiquement mis dans le champ *datDateOut* la date du jour et qu'il ne soit pas possible de saisir une commande future ou ayant plus de 15 jours:

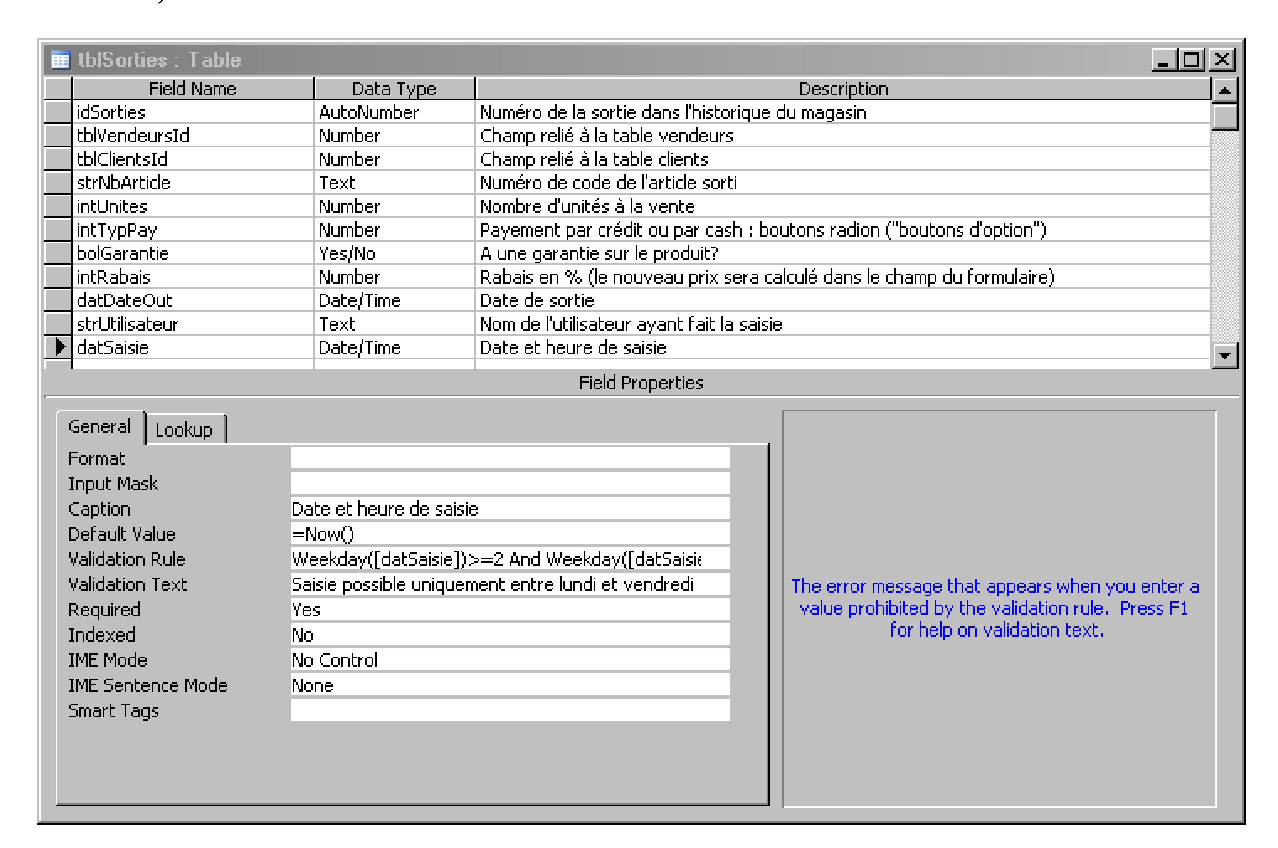


Remarque: Il n'est pas possible d'utiliser toute les formules disponibles dans MS Access dans les tables. Seulement les plus simples y sont autorisées (soit une très faible minorité)!

10.3 Fonctions Now comme valeur par défaut (tables)

Dans la table *tblSorties* nous souhaiterions qu'à chaque nouvelle saisie soit automatiquement mis dans un champ *datSaisie* la date et l'heure du jour de la saisie et que celle-ci soit valide que pendant les 5 jours ouvrés classiques de la semaine.

Pour cela, il suffit de mettre:



où la formule dans *Validation Rule* est (sur un PC avec les paramètres américains):

Weekday([datSaisie])>=2 And Weekday([datSaisie])<=6

Remarque: Il est impossible avec les formules traditionnelles (à ce jour) dans un champ de faire référence à un autre champ de la table!

10.4 Fonctions d'environnement (tables)

Signalons pour commencer deux fonctions très utiles dans les formulaires, requêtes, rapports ou tables:

=Environ("username")

qui renvoie le nom de l'utilisateur de la session. Ou si elle ne fonctionne pas:

=CurrentUser( )

Permet par exemple dans un formulaire de saisie de capter le nom d'utilisateur de la personne ayant fait la saisie, ou dans une requête de ne pas afficher les en fonction du nom de l'utilisateur (en mixant avec une fonction IIF).

Voici un exemple qui met automatiquement le nom de l'utilisateur lors de la saisie d'un nouveau champ et qui s'assure que celui-ci à une longueur d'au moins 5 caractères:

tblSorties : Table

Field Name	Data Type	Description
idSorties	AutoNumber	Numéro de la sortie dans l'historique du magasin
tblVendeursId	Number	Champ relié à la table vendeurs
tblClientsId	Number	Champ relié à la table clients
strNbArticle	Text	Numéro de code de l'article sorti
intUnites	Number	Nombre d'unités à la vente
intTypPay	Number	Payement par crédit ou par cash : boutons radion ("boutons d'option")
bolGarantie	Yes/No	A une garantie sur le produit?
intRabais	Number	Rabais en % (le nouveau prix sera calculé dans le champ du formulaire)
datDateOut	Date/Time	Date de sortie
strUtilisateur	Text	Nom de l'utilisateur ayant fait la saisie

Field Properties

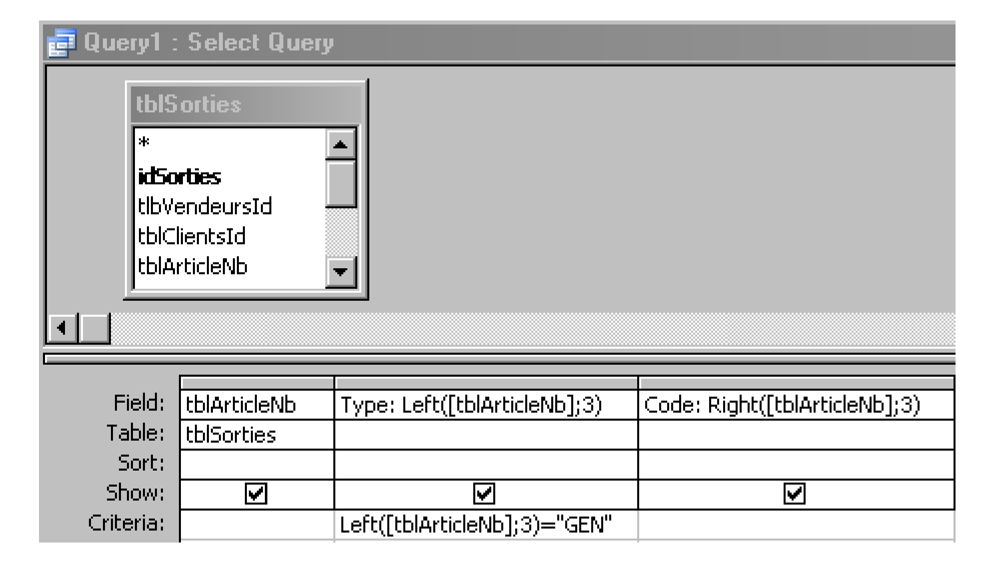
General | Lookup

Field Size	50
Format	
Input Mask	
Caption	Utilisateur
Default Value	=Environ("username")
Validation Rule	Len([strUtilisateur])>5
Validation Text	
Required	Yes
Allow Zero Length	No
Indexed	No
Unicode Compression	Yes
IME Mode	No Control
IME Sentence Mode	None
Smart Tags	

The error message that appears when you enter a value prohibited by the validation rule. Press F1 for help on validation text.

10.5 Fonction textes Left, Right, Instr (requête)

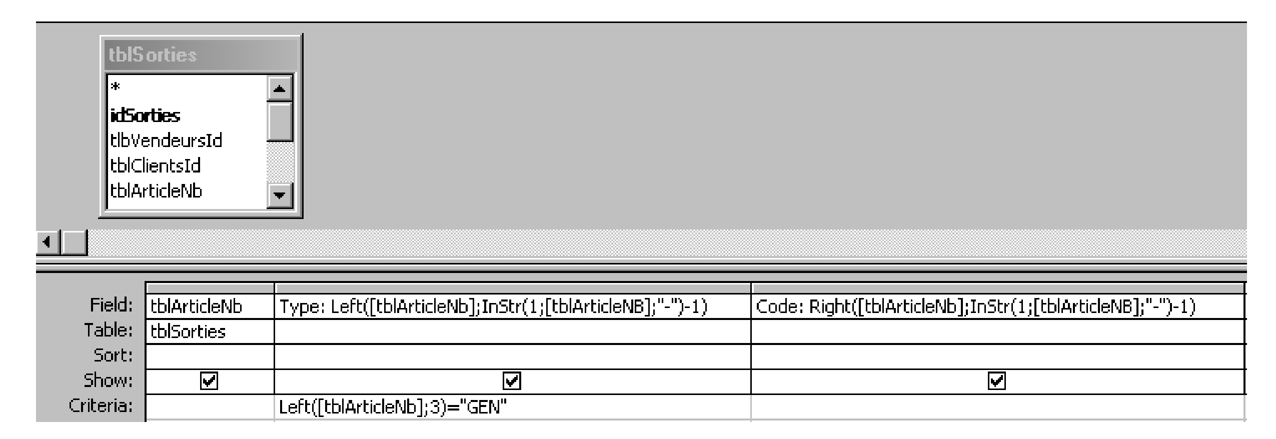
Nous souhaiterions arriver à une requête affichant dans une colonne le type et la catégorie de produit en se basant sur une décomposition du nom des articles de type GEN* vendus se trouvant dans la table *tblSorties*. Nous avons alors:



Ce qui donne:

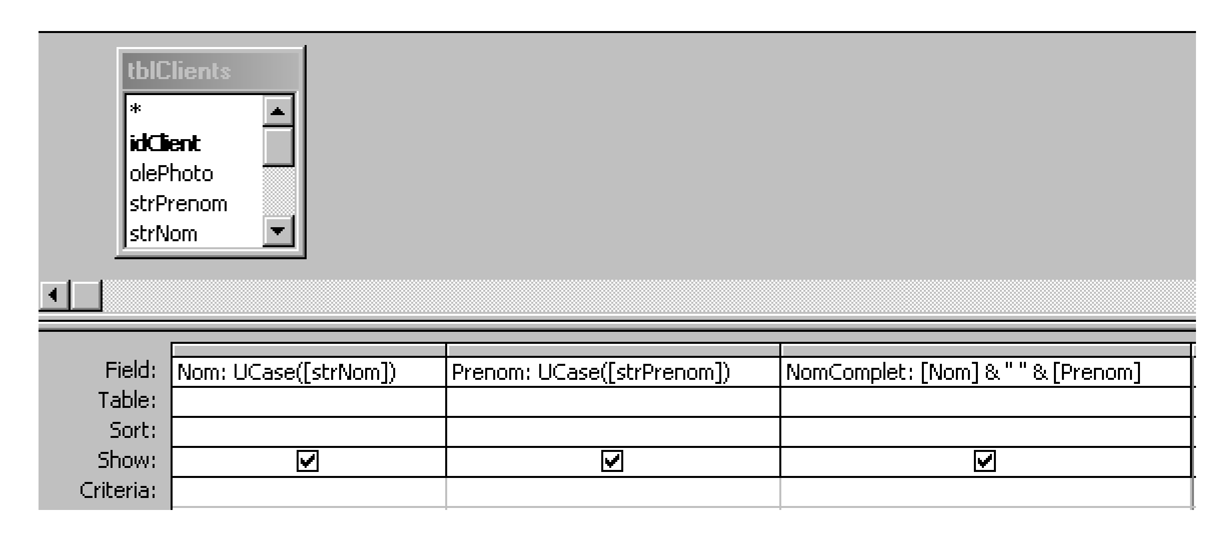
Code Article	Type	Code
GEN-001	GEN	001
GEN-003	GEN	003
GEN-002	GEN	002
GEN-003	GEN	003
GEN-002	GEN	002
GEN-004	GEN	004
GEN-003	GEN	003
GEN-006	GEN	006
GEN-001	GEN	001
GEN-004	GEN	004
GEN-003	GEN	003

Malheureusement il se peut que la longueur des codes articles ne soit pas toujours la même. Il faut alors utiliser une fonction très courante dans MS Access qui est capable de déterminer la position du symbole de séparation. Nous aurons alors:



10.6 Fonctions textes UCase, LCase, &, TRIM, StrConv (requête)

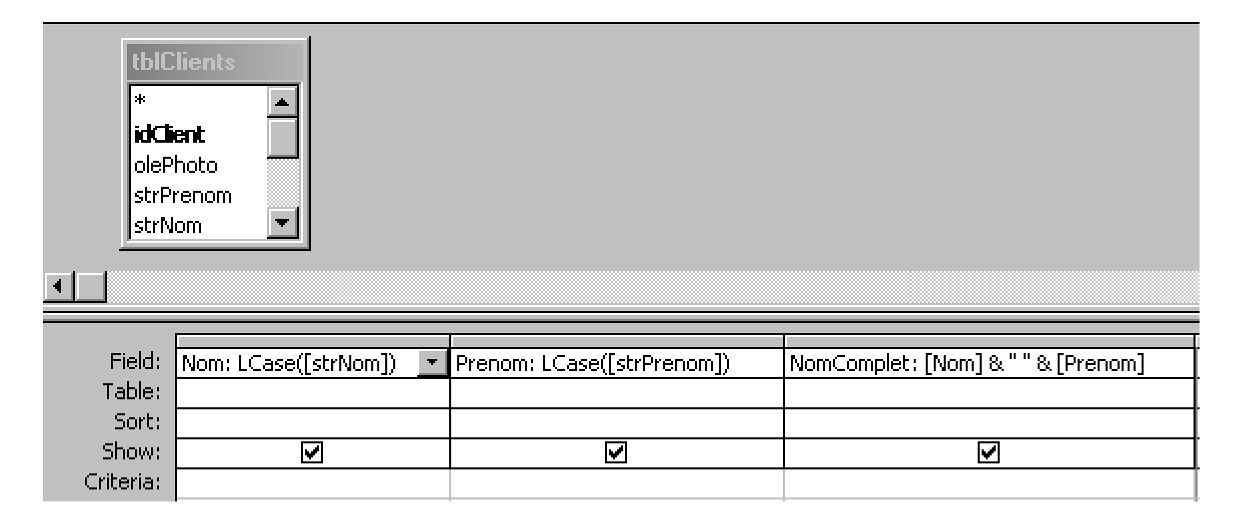
Si nous souhaitons prendre plusieurs champs de texte, transformer tous les caractères en majuscules et ensuite les concaténer il suffit d'utiliser *UCase* et *&* comme dans MS Excel:



ce qui donnera:

	Nom	Prenom	NomComplet
▶	DUTRON	ALAIN	DUTRON ALAIN
	ANGEL	CHARLIE	ANGEL CHARLIE
	BOLTZMANN	ALBERT	BOLTZMANN ALBERT
*			

et respectivement si nous voulons tout en minuscules:



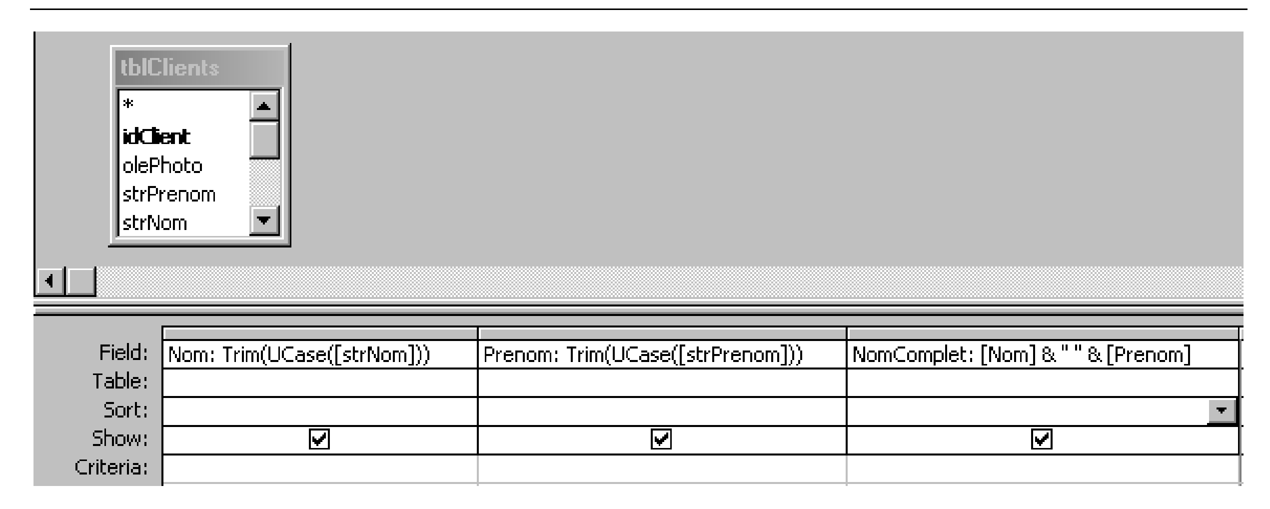
ce qui donnera:

	Nom	Prenom	NomComplet
▶	DUTRON	ALAIN	DUTRON ALAIN
	ANGEL	CHARLIE	ANGEL CHARLIE
	BOLTZMANN	ALBERT	BOLTZMANN ALBERT
*			

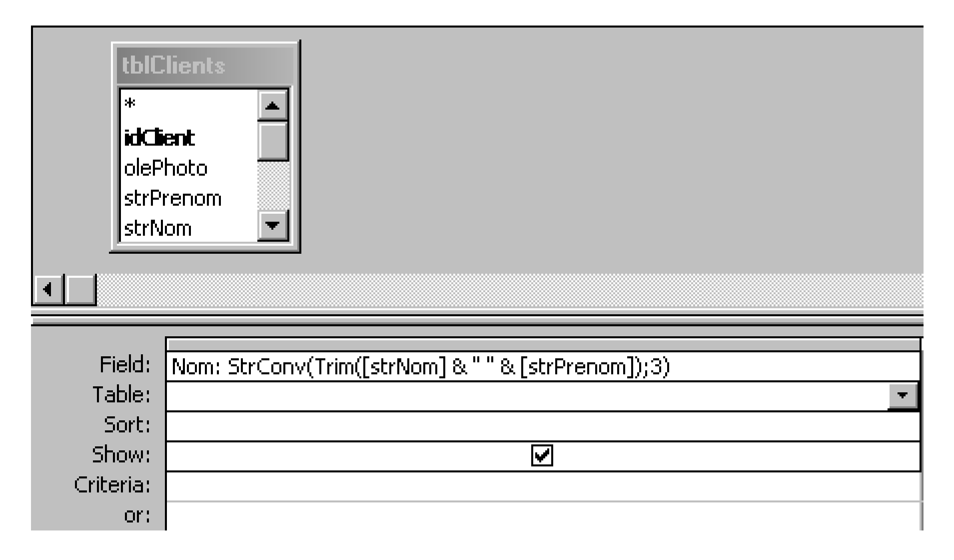
Il arrive malheureusement (environ 2 à 3% des données) que les utilisateurs saisissent des espaces à doubles dans des noms divers. Ce qui donne alors des catastrophes du genre:

	Nom	Prenom	NomComplet
▶	dutron	alain	dutron alain
	angel	charlie	angel charlie
	boltzmann	albert	boltzmann albert
*			

Il faut alors souvent nettoyer la sortie des requêtes en utilisant la fonction *TRIM*:



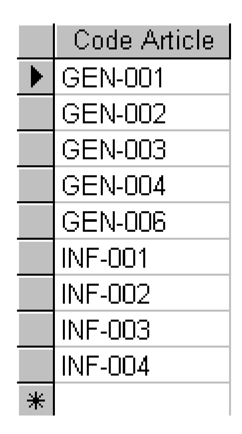
Si on souhaite mettre uniquement la première lettre de chaque mot en majuscule il faudra utiliser la fonction *StrConv*:



où le paramètre 3 signifie que l'on souhaite que la première lettre de chaque mot en majuscule (la valeur 1 correspond à *UCase* et la valeur 2 à *LCase*).

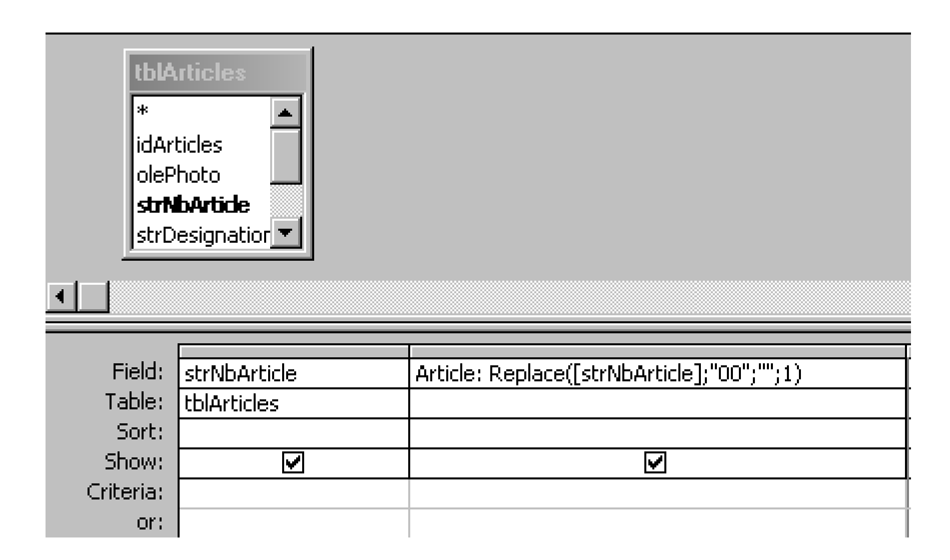
10.7 Fonctions replace et MID (requête)

Considérons la requête retournant les éléments suivants:



Code Article
GEN-001
GEN-002
GEN-003
GEN-004
GEN-006
INF-001
INF-002
INF-003
INF-004

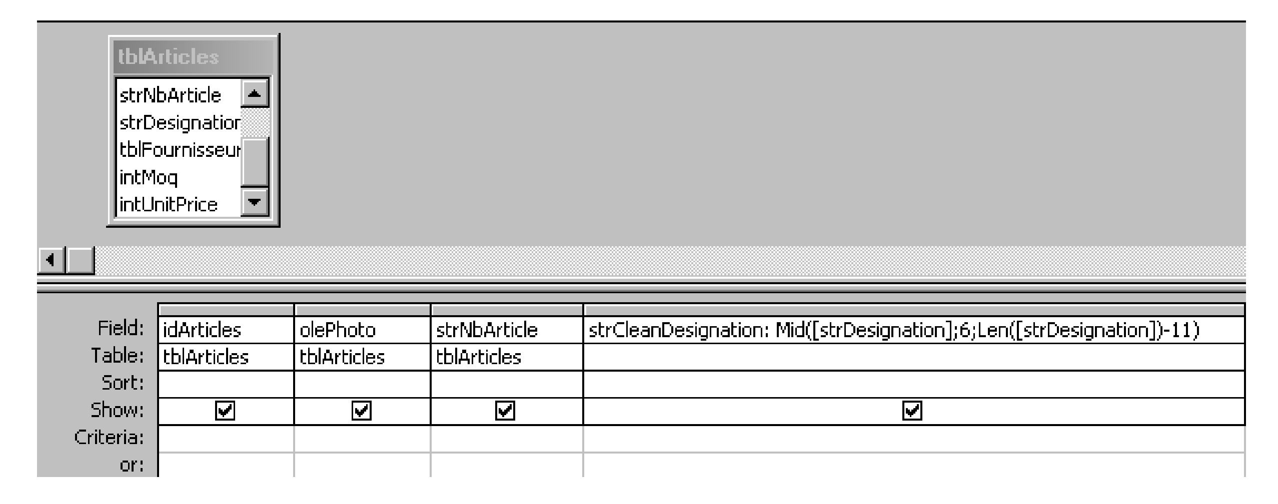
Nous souhaiterions remplacer tous les double zéro (00) par rien. Nettoyer des textes dans MS Access arrive souvent dès que l'on travaille avec SAP ou le Web. La fonction *Replace* est alors très utile dans ce contexte:



De même que la fonction *Replace*, la fonction *Mid* est elle aussi souvent utilisée pour nettoyer des chaînes de caractères ou prendre qu'une partie de ceux-ci. Voyons un exemple avec la table *tblArticles* dont le champ de *Description* contient des caractères parasites provenant d'un import du web:

idArticles	Photo	Article	Description	Fournisseur	M.O.Q.	Prix à l'unité
1		GEN-001	Crayons	1	100	CHF. 0.21
2		GEN-002	Enveloppes (10 pces)	2	50	CHF. 0.53
3		GEN-003	DIN A4 Papier (500 feuilles)	3	50	CHF. 25.04
4		GEN-004	Post-It Notes 656	1	60	CHF. 10.29
9		GEN-006	Stylos rouges	1	100	CHF. 0.53
6		INF-001	Tambour	3	5	CHF. 261.45
7		INF-002	Disquette (3.5')	3	1000	CHF. 1.37
8		INF-003	Etiquettes LASER (25 feuilles)	3	10	CHF. 37.70
10		INF-004	oner	3	20	CHF. 90.20
(AutoNumber)				0		CHF. 0.00

Même s'il existe plusieurs manière d'arriver au même résultat, en utilisation la fonction *Mid* et la requête suivante:

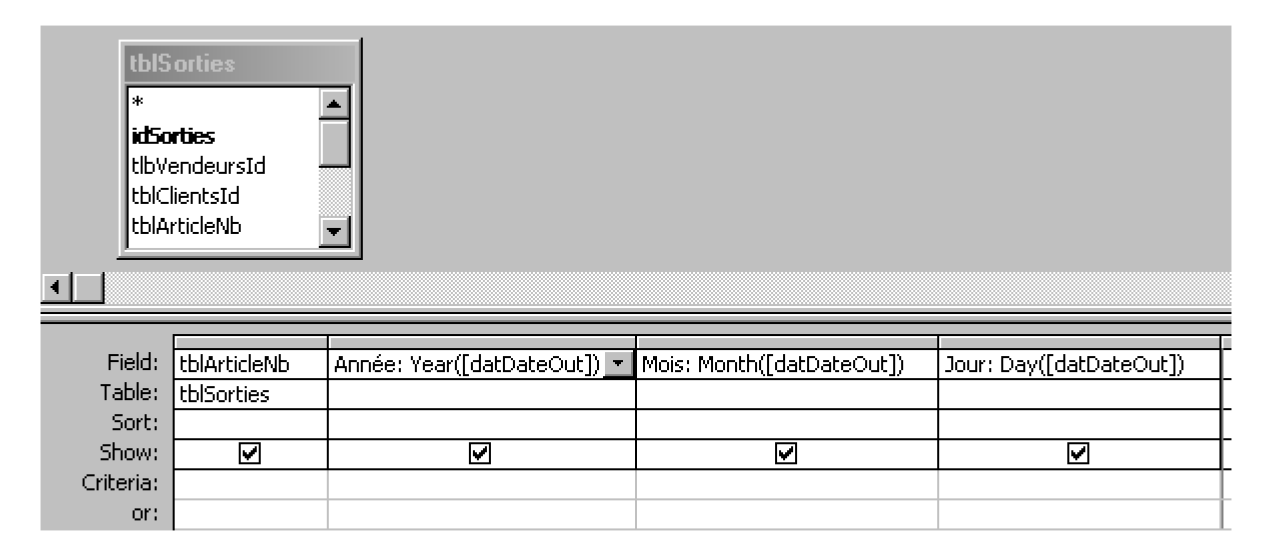


ce qui donnera:

	idArticles	Photo	Article	strCleanDesignation	Fournisseur	M.O.Q.	Prix à l'unité
▶	1		GEN-001	Crayons	1	100	CHF. 0.21
	2		GEN-002	Enveloppes (10 pces)	2	50	CHF. 0.53
	3		GEN-003	DIN A4 Papier (500 feuilles)	3	50	CHF. 25.04
	4		GEN-004	Post-It Notes 656	1	60	CHF. 10.29
	9		GEN-006	Stylos rouges	1	100	CHF. 0.53
	6		INF-001	Tambour	3	5	CHF. 261.45
	7		INF-002	Disquette (3.5')	3	1000	CHF. 1.37
	8		INF-003	Etiquettes LASER (25 feuilles)	3	10	CHF. 37.70
	10		INF-004	Toner	3	20	CHF. 90.20
*	(AutoNumber)				0		CHF. 0.00

10.8 Fonctions de dates Year, Day, Month (requête)

Si nous souhaitons décomposer les années, mois et jour d'un champ de date dans une requête pour faire des analyses particulières il suffit d'utiliser les trois fonctions mentionnées dans le titre:

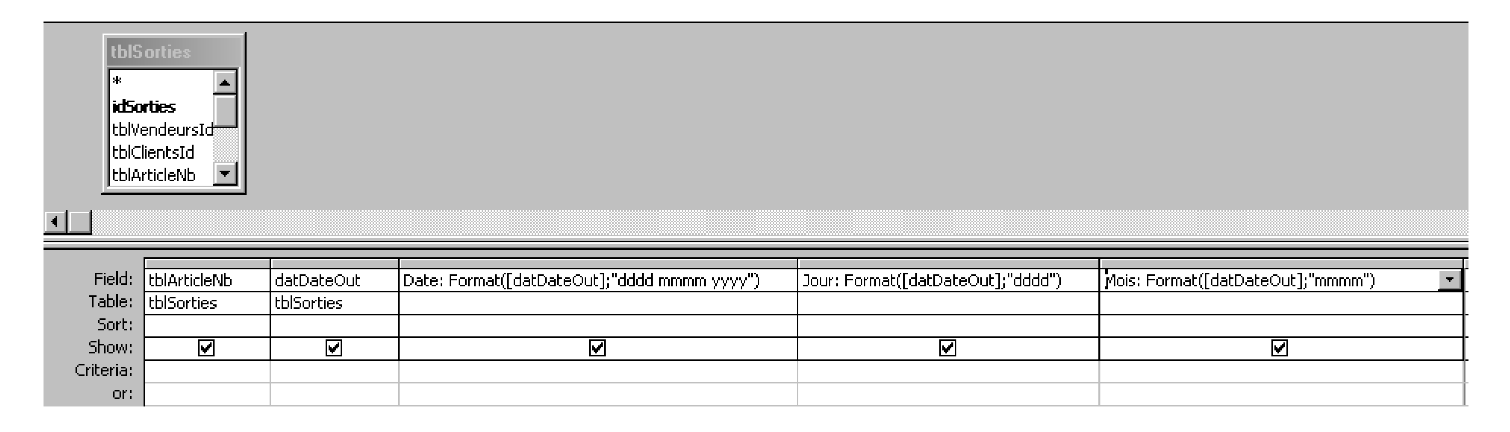


ce qui donne immédiatement:

Code Article	Année	Mois	Jour
GEN-001	1996	11	28
GEN-003	1996	12	10
INF-002	1996	12	10
GEN-002	1996	12	11
GEN-003	1996	12	11
INF-002	1996	12	11
INF-003	1996	12	11
GEN-002	1996	12	18
INF-001	1996	12	18
INF-003	1996	12	18
GEN-004	1996	12	30
INF-001	1996	12	30
INF-003	1996	12	30
GEN-003	1997	1	2
GEN-006	1997	1	2
INF-002	1997	1	2
GEN-001	1997	1	8

10.9 Fonction de formatage de dates et DatePart (requête)

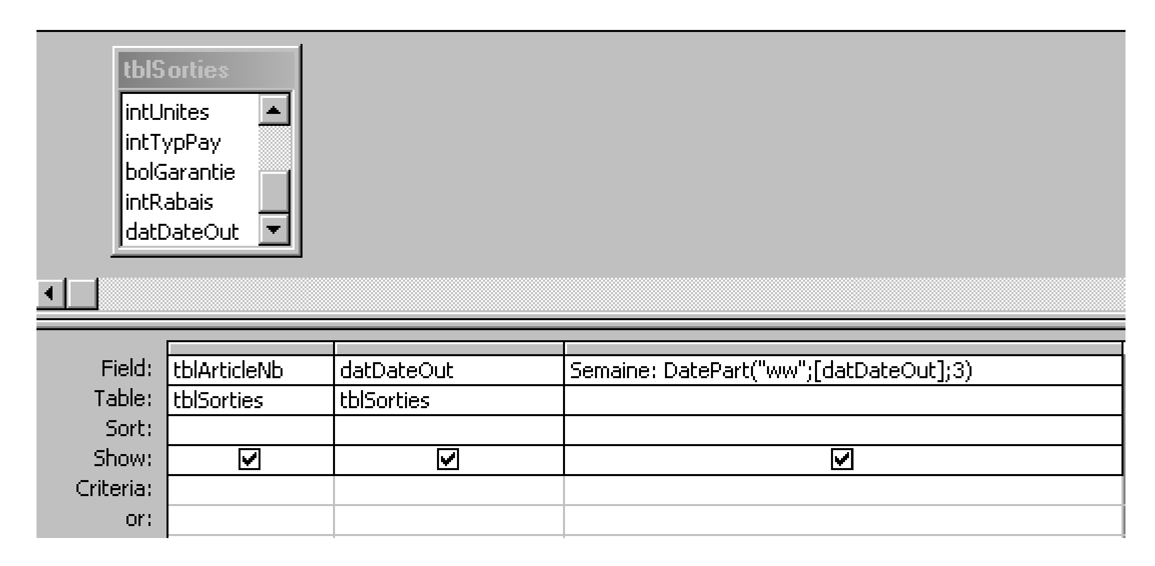
Dans le même genre:



ce qui donne:

Code Article	Date sortie	Date	Jour	Mois
GEN-001	28.11.1996	jeudi novembre 1996	jeudi	novembre
GEN-003	10.12.1996	mardi décembre 1996	mardi	décembre
INF-002	10.12.1996	mardi décembre 1996	mardi	décembre
GEN-002	11.12.1996	mercredi décembre 1996	mercredi	décembre
GEN-003	11.12.1996	mercredi décembre 1996	mercredi	décembre
INF-002	11.12.1996	mercredi décembre 1996	mercredi	décembre
INF-003	11.12.1996	mercredi décembre 1996	mercredi	décembre
GEN-002	18.12.1996	mercredi décembre 1996	mercredi	décembre
INF-001	18.12.1996	mercredi décembre 1996	mercredi	décembre
INF-003	18.12.1996	mercredi décembre 1996	mercredi	décembre
GEN-004	30.12.1996	lundi décembre 1996	lundi	décembre
INF-001	30.12.1996	lundi décembre 1996	lundi	décembre
INF-003	30.12.1996	lundi décembre 1996	lundi	décembre
GEN-003	02.01.1997	jeudi janvier 1997	jeudi	janvier
GEN-006	02.01.1997	jeudi janvier 1997	jeudi	janvier
INF-002	02.01.1997	jeudi janvier 1997	jeudi	janvier
GEN-001	08.01.1997	mercredi janvier 1997	mercredi	janvier
GEN-004	08.01.1997	mercredi janvier 1997	mercredi	janvier
INF-002	08.01.1997	mercredi janvier 1997	mercredi	janvier
GEN-003	15.01.1997	mercredi janvier 1997	mercredi	janvier

ou avec la fonction *DatePart* on peut avoir les numéros de semaines en commençant par la première semaine complète de l'année (paramètre 3):

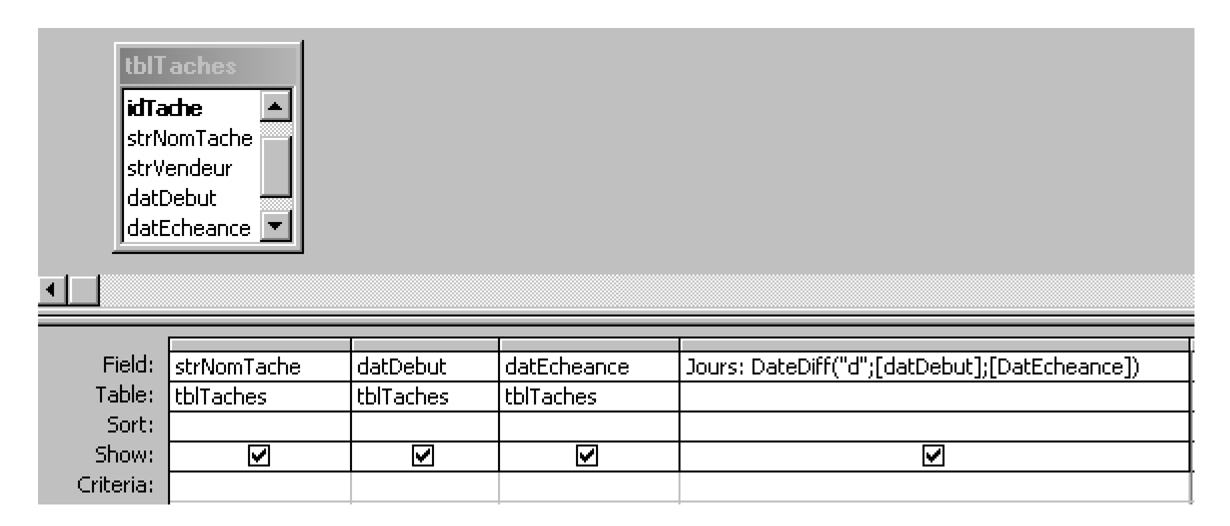


Ce qui donne:

Code Article	Date sortie	Semaine
GEN-001	28.11.1996	49
GEN-003	10.12.1996	51
INF-002	10.12.1996	51
GEN-002	11.12.1996	51
GEN-003	11.12.1996	51
INF-002	11.12.1996	51
INF-003	11.12.1996	51
GEN-002	18.12.1996	52
INF-001	18.12.1996	52
INF-003	18.12.1996	52
GEN-004	30.12.1996	53
INF-001	30.12.1996	53
INF-003	30.12.1996	53
GEN-003	02.01.1997	1
GEN-006	02.01.1997	1
INF-002	02.01.1997	1
GEN-001	08.01.1997	2
GEN-004	08.01.1997	2
INF-002	08.01.1997	2

10.10 Fonction de date DateDiff (requête)

Lorsque l'on a une base de données permettant de gérer des tâches ou projets il est très fréquent de devoir calculer le nombre de jours entre deux dates. Nous avons alors avec la table *tblTaches*:

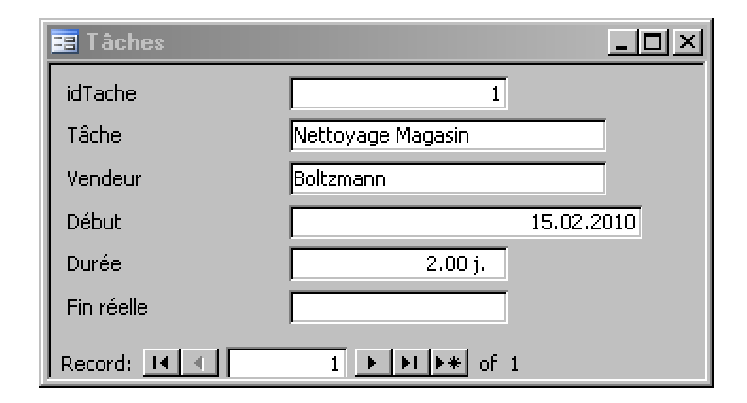


et il est possible bien évidemment de changer le "d" par un "m" pour mois et un "y" pour les années.

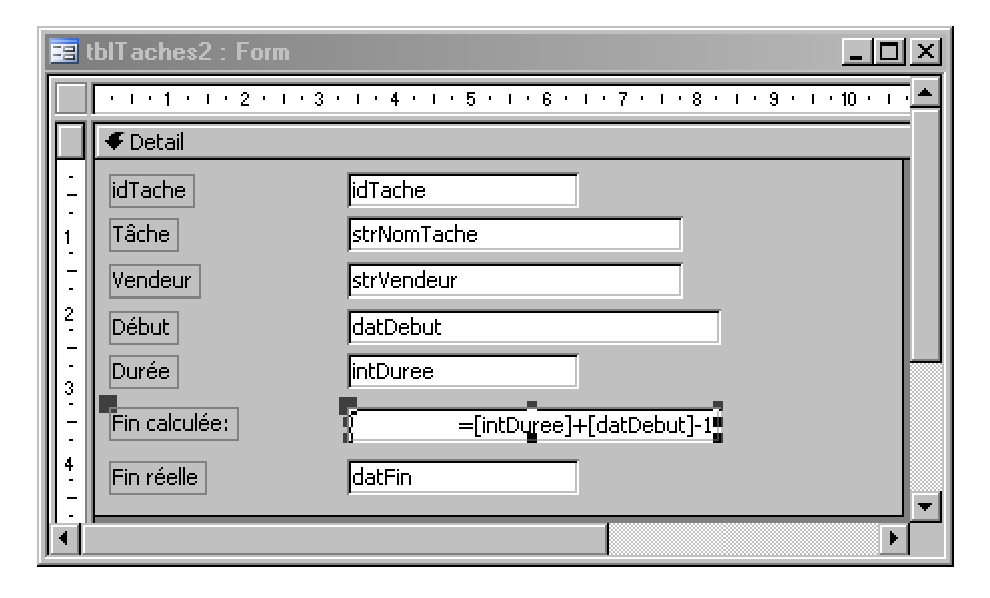
Remarque: Malheureusement il n'existe pas à ce jour de fonction ne prenant pas en compte le Week-End. Il faut passer par du VBA.

10.11 Calculs de jours avec DateAdd (Formulaire)

Si vous avez une table de tâches et un formulaire de saisie pour celle du type suivant:

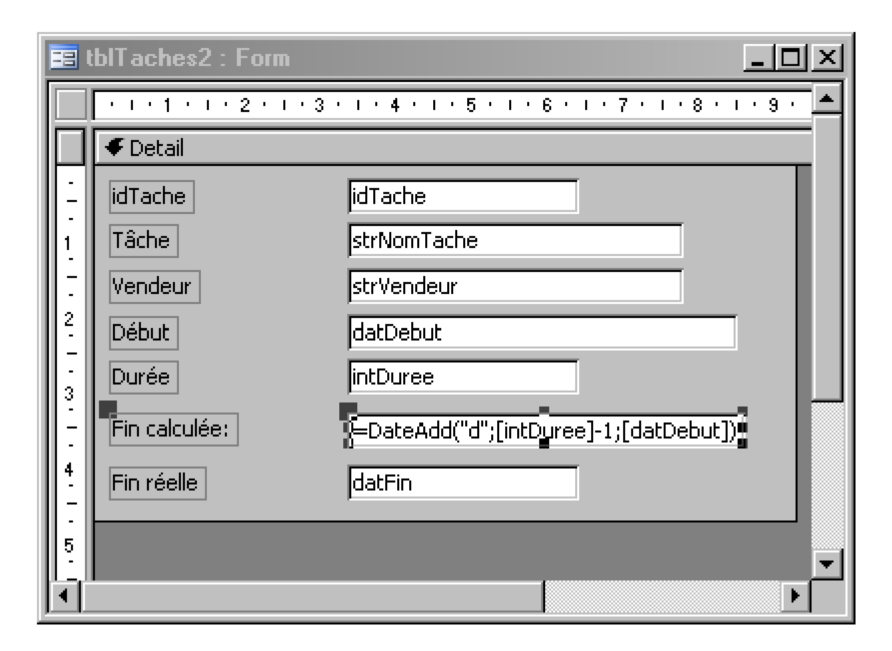


Vous souhaiteriez pouvoir ajouter un champ avec une formule qui indique la date de fin calculée week-end compris. Il suffit alors de créer un champ du type:



mais si vous souhaitez que les week-end ne soient pas pris en compte il faudra passer par du VBA.

On peut arriver au même résultat avec la fonction DateAdd:

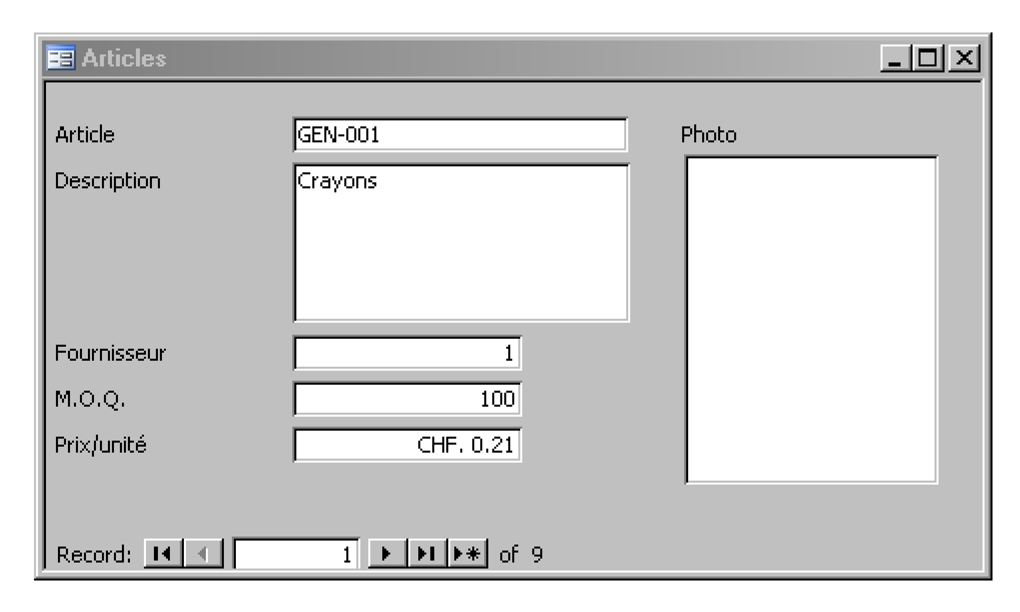


avec par contre l'avantage que cette fonction *DateAdd* est capable de prendre plusieurs types d'unités différentes dans ses calculs:

Valeur	Correspondance
yyyy	Années
q	Trimestres
m	Mois
d	Jours
h	Heures
n	Minutes
s	Secondes

10.12 Fonction Logique IIf (formulaire)

Nous allons voir maintenant l'équivalent de la fonction SI connue dans MS Excel mais en version MS Access. Pour cela, nous allons créer un champ dans le formulaire *frmArticles*:



qui affiche "Trop cher" si le prix à l'unité est supérieur à 100.- ou "OK" lors que le prix en est inférieur.



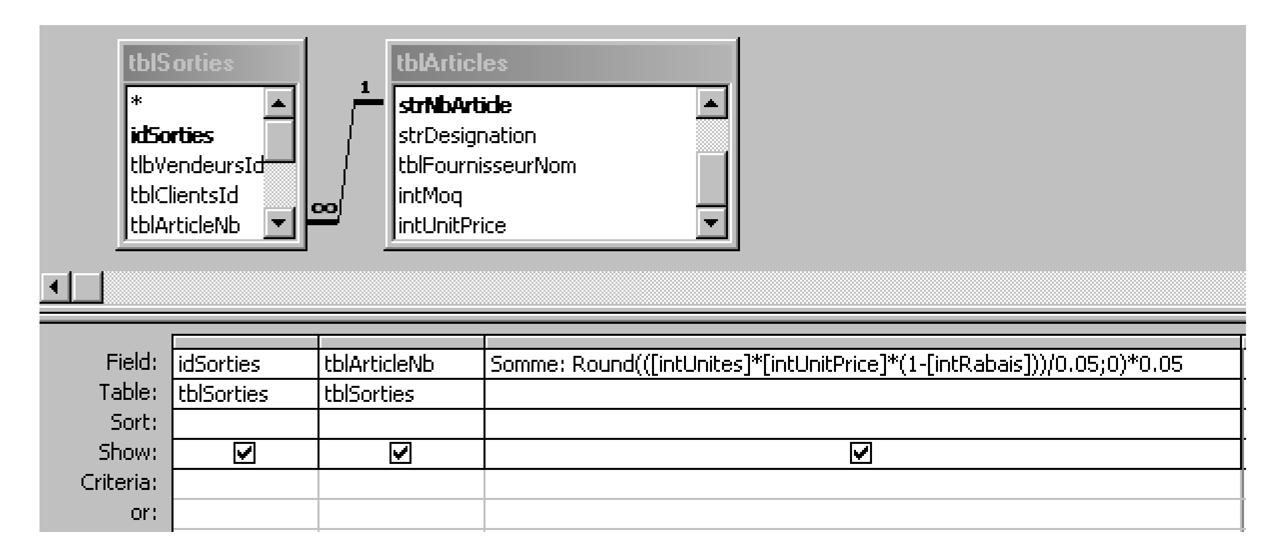
Remarque: La fonction SI dans MS Access est nommée *VraiFaux* en français ou *IIF* en anglais.

Nous souhaitons cependant afficher "Trop cher" que si le prix est supérieur à 100.- et que la M.O.Q. aussi. Dans ce cas, à nouveau nous nous retrouvons avec une syntaxe presque identique à celle de MS Excel:

Info Prix: =IIf([intUnitPrice]>100 And [intMoq]>100;"Trop Cher";"OK")

10.13 Fonctions d'arrondi Round (requête)

MS Access ne comporte pas les possibilités d'arrondis de MS Excel si on veut arrondir aux 5 centimes les plus proches il faudra alors utiliser la veille formule suivante:



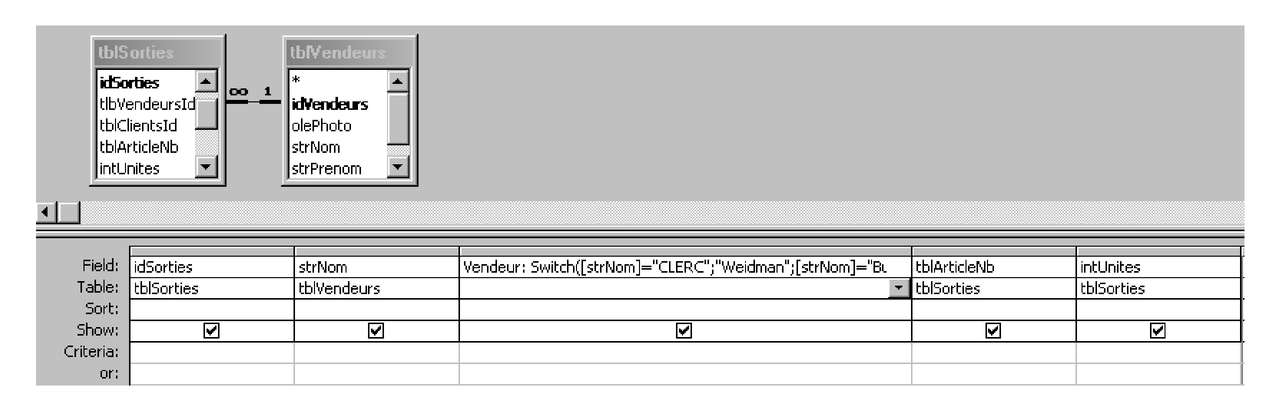
10.14 Fonction conditionnelle Switch (requête)

Lorsque le développeur d'une base de données MS Access a besoin de faire une multitude de IIF imbriquées les uns dans les autres il se retrouve vitre confronté à une fonction illisible à l'écran. Il vaut alors mieux dans les cas simples utiliser la fonction *Switch*.

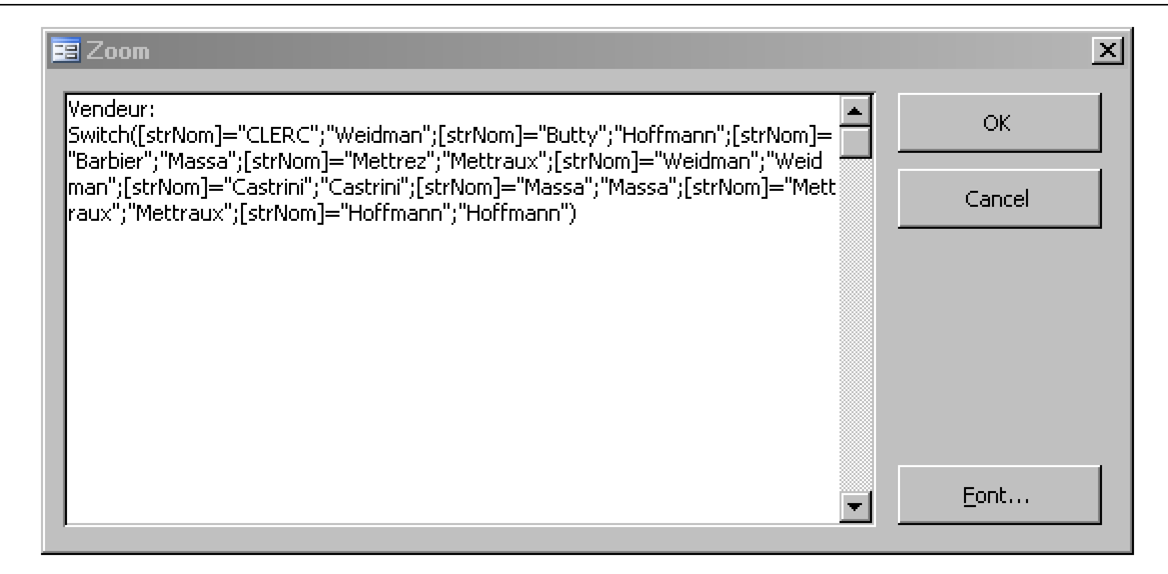
Considérons la requête suivante sortant le résultat suivant:

idSorties	Vendeurs	Code Article	Unités vendues
1	Weidman	GEN-001	50
2	Castrini	GEN-003	10
4	Clerc	INF-002	1500
5	Butty	GEN-002	100
6	Mettraux	GEN-003	20
8	Massa	INF-002	500
9	Hoffmann	INF-003	30
10	Clerc	GEN-002	50
11	Massa	INF-001	5
12	Mettrez	INF-003	10
13	Barbier	GEN-004	80
14	Butty	INF-001	15
15	Clerc	INF-003	5
16	Castrini	GEN-003	30
17	Weidman	GEN-006	200
18	Mettraux	INF-002	750
19	Hoffmann	GEN-001	100
20	Castrini	GEN-004	30
21	Clerc	INF-002	100
22	Massa	GEN-003	30
23	Mettrez	INF-002	20
24	Butty	INF-003	750
25	Barbier	GEN-002	1

Nous souhaiterions y remplacer *Clerc* par *Weidman*, *Butty* par *Hoffmann*, *Barbier* par *Massa* et *Mettrez* par *Mettraux* et les autres devront rester tels qu'ils sont. Cela donne dès lors dans la requête avec la fonction *Switch*:



Soit avec un zoom sur la fonction:

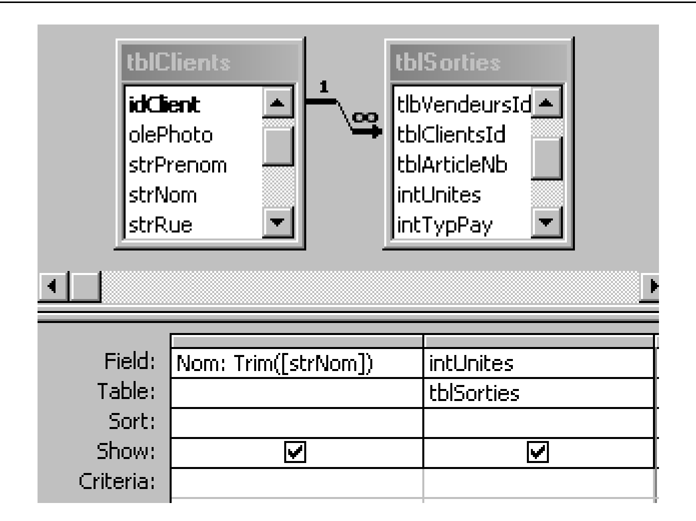


ce qui donne au final:

idSorties	Nom	Vendeur	Code Article	Unités vendues
1	WEIDMAN	Weidman	GEN-001	50
17	WEIDMAN	Weidman	GEN-006	200
5	BUTTY	Hoffmann	GEN-002	100
14	BUTTY	Hoffmann	INF-001	15
24	BUTTY	Hoffmann	INF-003	750
4	CLERC	Weidman	INF-002	1500
10	CLERC	Weidman	GEN-002	50
15	CLERC	Weidman	INF-003	5
21	CLERC	Weidman	INF-002	100
2	CASTRINI	Castrini	GEN-003	10
16	CASTRINI	Castrini	GEN-003	30
20	CASTRINI	Castrini	GEN-004	30
8	MASSA	Massa	INF-002	500
11	MASSA	Massa	INF-001	5
22	MASSA	Massa	GEN-003	30
12	METTREZ	Mettraux	INF-003	10
23	METTREZ	Mettraux	INF-002	20
13	BARBIER	Massa	GEN-004	80
25	BARBIER	Massa	GEN-002	1
6	METTRAUX	Mettraux	GEN-003	20
18	METTRAUX	Mettraux	INF-002	750
9	HOFFMANN	Hoffmann	INF-003	30
19	HOFFMANN	Hoffmann	GEN-001	100
(AutoNumber)				

10.15 Fonctions IsNull et NZ (requête)

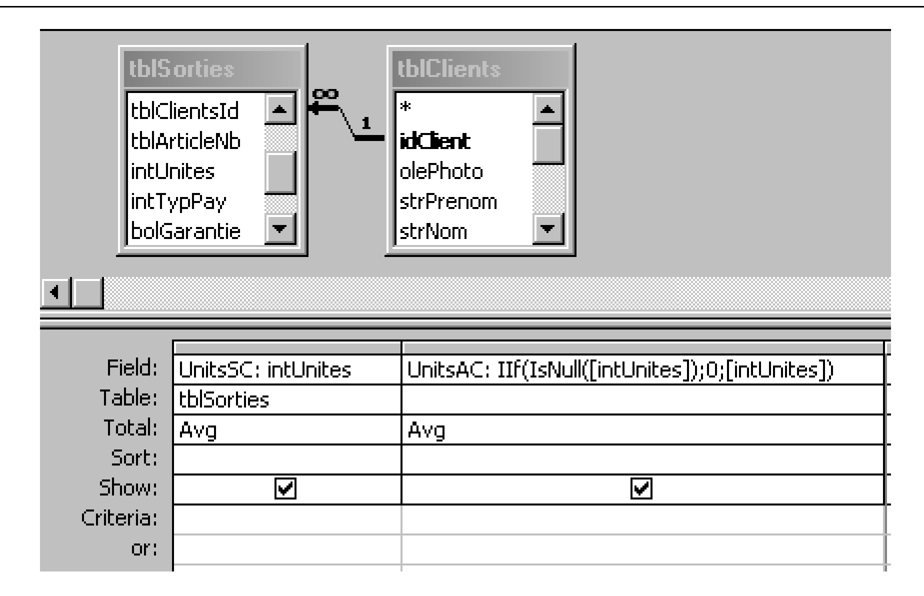
Lorsque vous effectuez une requête avec jointure vous aurez des champs avec des valeurs nulles comme la montre la requête ci-dessous:



qui donne lorsqu'un client de la table *tblClients* n'a jamais fait d'achats (voir les deux dernières lignes):

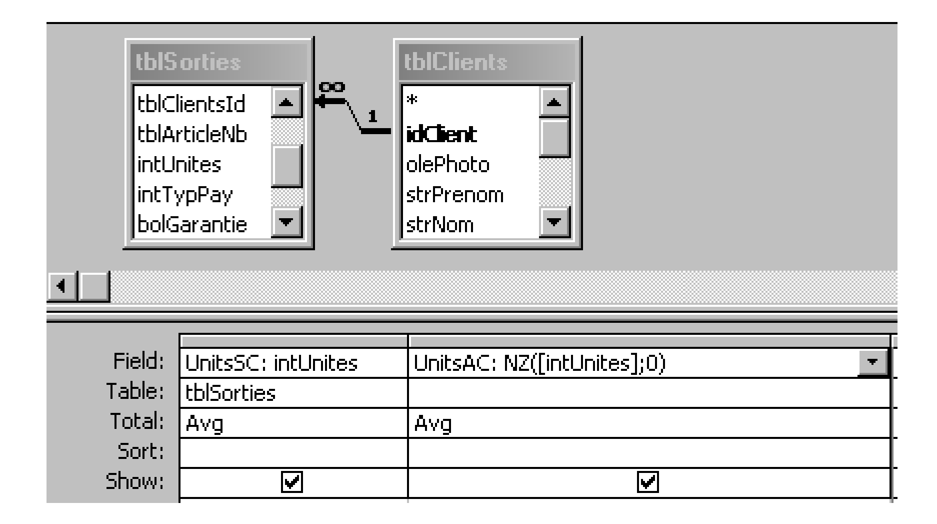
Nom	Unités vendues
Dutron	1500
Dutron	100
Dutron	30
Dutron	15
Dutron	30
Dutron	750
Dutron	750
Angel	10
Angel	20
Angel	5
Angel	10
Angel	80
Angel	200
Angel	100
Angel	30
Angel	1
Boltzmann	50
Boltzmann	500
Boltzmann	50
Boltzmann	5
Boltzmann	100
Boltzmann	30
Boltzmann	20
Isoz	
Jobs	

Or si nous souhaitons une moyenne des unités vendues comprenant TOUS les clients et une moyenne comprenant SEULEMENT les clients ayant fait des achats il nous faudra utiliser la fonction IsNull, NZ ou Null:



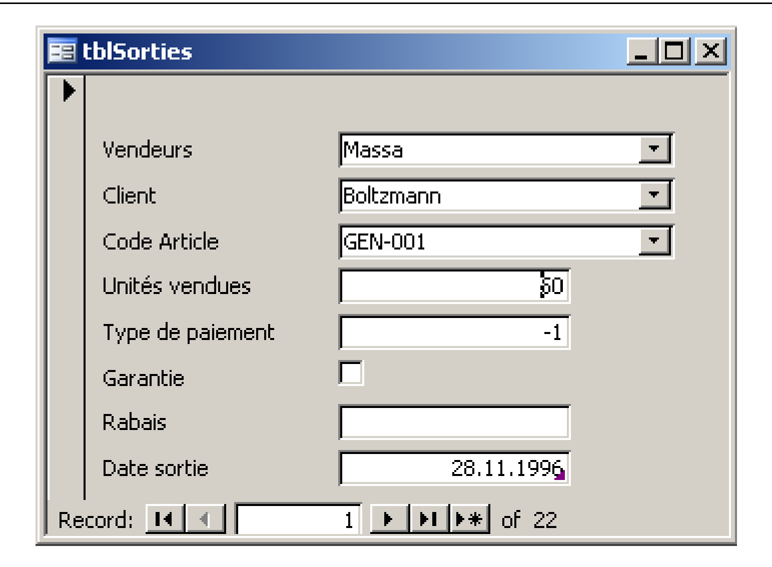
où la deuxième colonne permet donc de faire une moyenne sur TOUS les clients!

Une deuxième solution qui donne exactement le même résultat:



10.16 Fonction DSum (formulaire)

Considérons le formulaire (pas fini!) des sorties des articles ci-dessous:



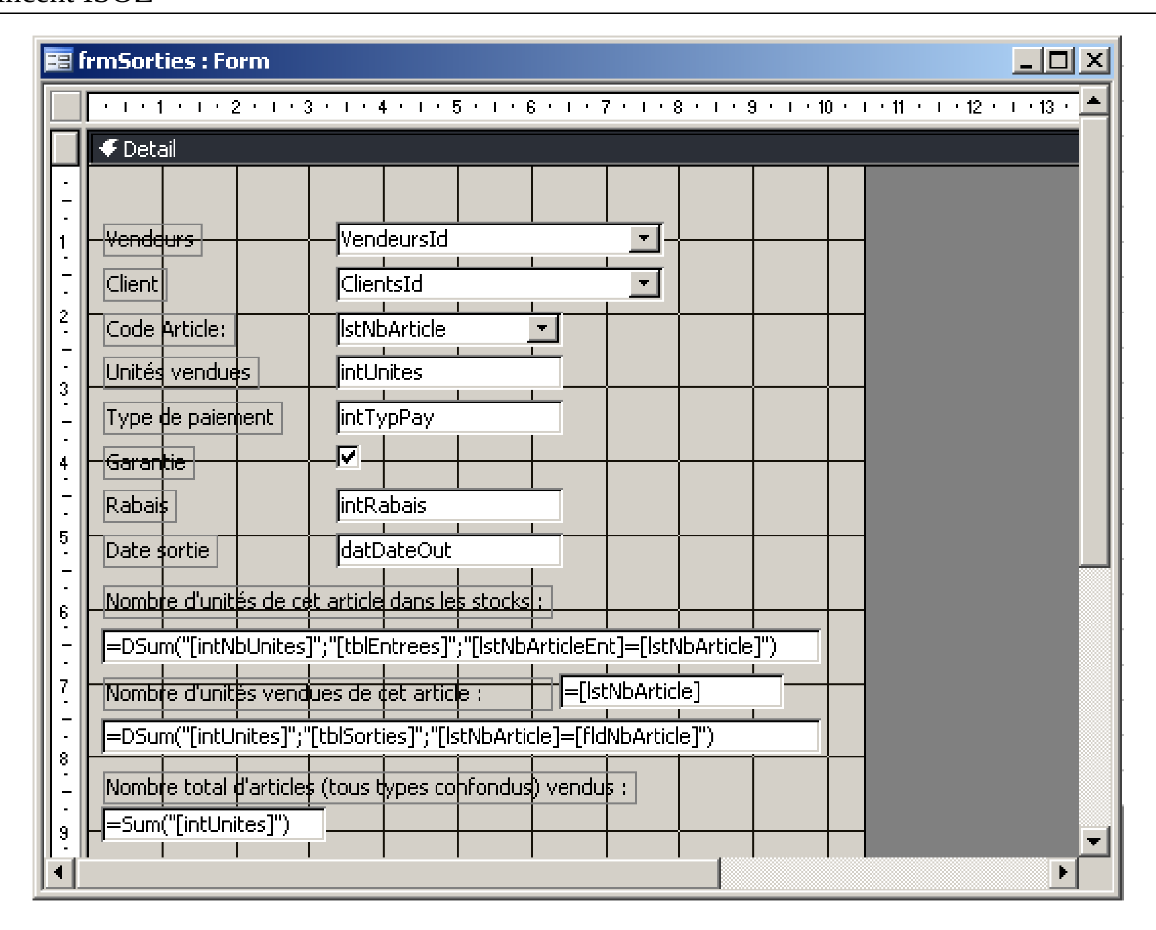
Les questions sont les suivantes:

1. Comment afficher toute la quantité du nombre d'unités vendue de tous les types d'articles

2. Comment afficher dans ce formulaire pour un article donné le nombre total d'unités vendues ?

3. Idem que (2) mais le nombre restant ?

Pour les autres, l'idée va consister à utiliser la fonction *DSum* (en anglais) et *Sum* de la manière suivante dans des champs que vous aurez créé (vous allez voir que la manière dont nous avons nommé les champs peut porter à confusion d'où l'importance d'une stratégie de nommage):



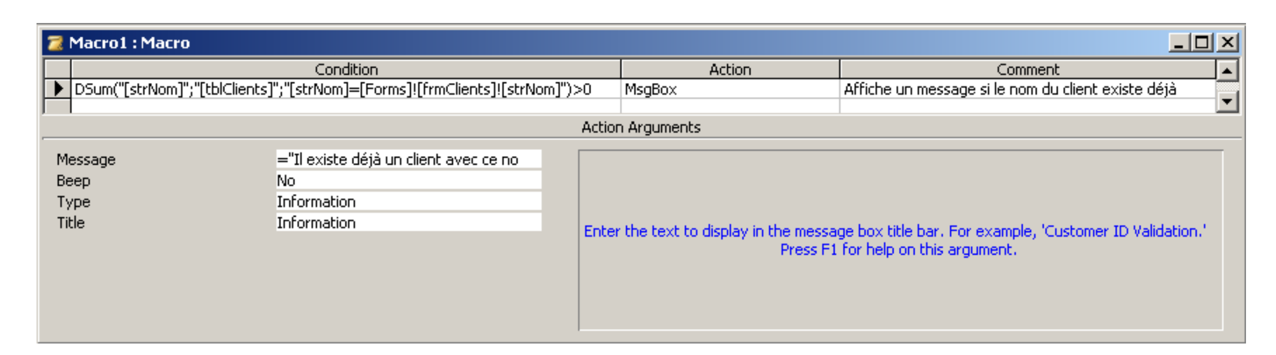
et il est bien évidemment possible d'utiliser d'autres fonctions que *Sum* pour faire des statistiques de la même table que celle sur laquelle est principalement basé le formulaire.

10.17 Fonction DCount (formulaire)

La fonction *DCount* a une syntaxe en tout point similaire à la fonction *DSum*. Elle compte simplement le nombre d'enregistrements au lieu de sommer leur valeur.

Elle peut être très pratique dans le cadre des macros. Effectivement, il est relativement aisé dans le formulaire de saisie des nouveaux clients ou vendeurs de faire en sorte que lorsque l'utilisateur quitte le champ *strNom*, la macro contrôle s'il existe déjà un vendeur du même nom (bon on peut faire pareil avec les index multiples mais l'idée peut être développée dans d'autres cas d'applications).

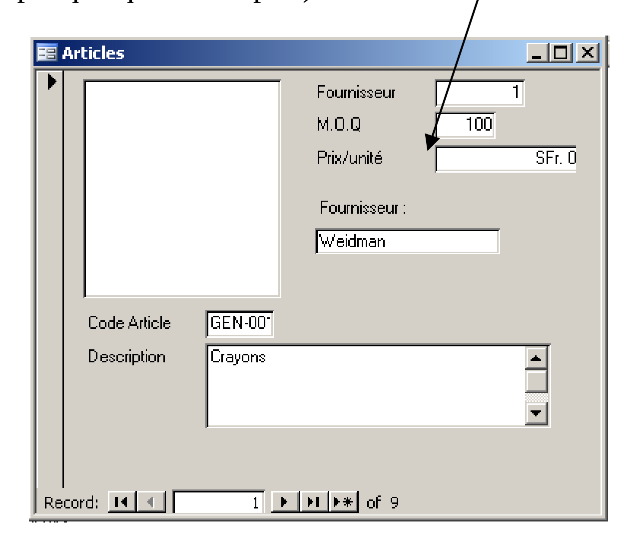
Un exemple qui peut être fait avec votre formateur dans le cadre du formulaire de saisie de nouveaux clients:



10.18 Fonction DLookUp (formulaire)

Petit exercice de style… et très demandé (ainsi que découverte d'une des fonctions les plus puissantes de MS Access):

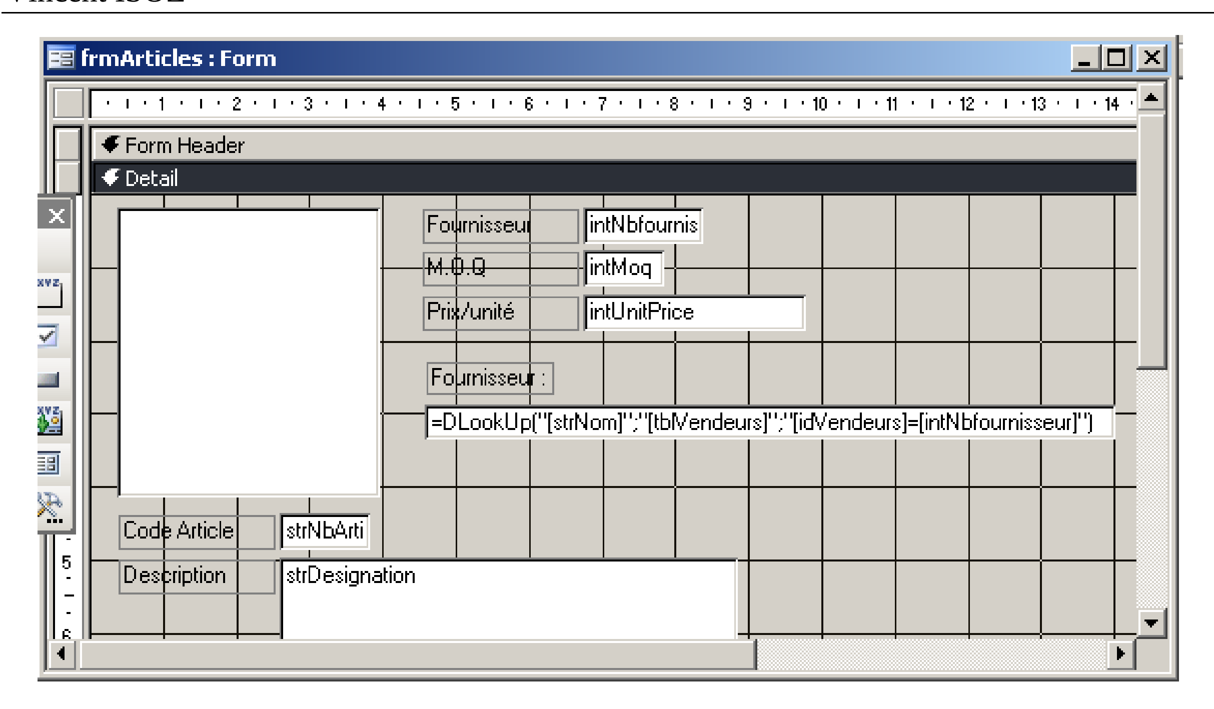
Comment afficher (par exemple) sur le formulaire de saisie d'articles, le plus simplement et esthétiquement possible, pour chaque article le nom du fournisseur (et non pas l'id !) correspondant à un article donné tel que présenté ci-dessous (et pas autrement mis à part l'emplacement du champ à quelque chose prêt):



Essayez tout d'abord avec l'assistant de formulaire de trouver le résultat…

Remarque: Si vous avez déjà le nom du fournisseur (car cela dépend de votre choix antérieur lors du cours de base où nous faisons exprès de créer des erreurs !) faites en sorte d'avoir le délai de livraison qui s'affiche sur le formulaire en fonction du nom du fournisseur.

La solution (il faut bien que vous l'ayez quelque part mais ne "trichez" pas en passant tout de suite à celle-ci sinon vous ne verrez pas l'intérêt de l'exercice) consiste à utiliser la très fameuse fonction DLOOKUP (RechDom en français) de MS Access et dans un nouveau champ, d'écrire la fonction suivante:



Faites de sortes que toutes les "champs calculés" soient verrouillées et inactifs afin que l'utilisateur ne puisse pas cliquer dedans (c'est une opération qu'il faut aussi savoir faire dans les formulaires ressortant de requêtes comme nous le verrons plus tard).

En tant qu'exercice, refaites de même que précédemment mais avec le délai de livraison (c'est plus pertinent) ainsi qu'avec le canton de localisation du fournisseur et le pays.

Petite remarque ! Rappelez-vous qu'au cours de base nous avions fait exprès de mettre les clés primaires au mauvais endroit dans les tables *tblPays* et *tblCantons*. Nous vous demandons dès lors la chose suivante afin de vous exercer à nouveau avec les concepts de clés primaires:

1. Pour le canton, ne changez rien

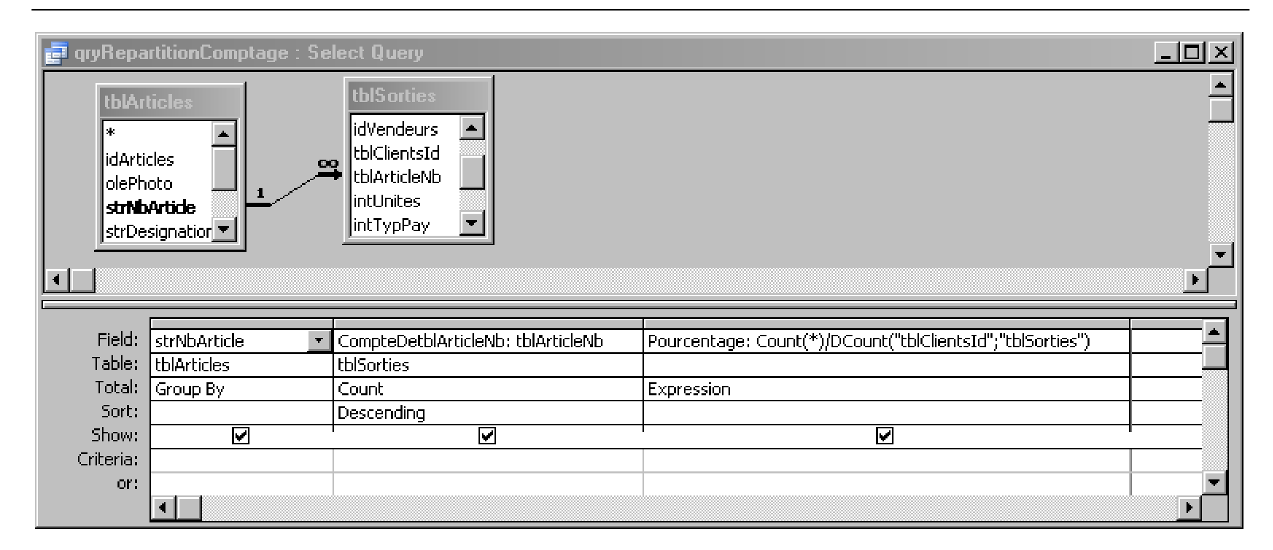
2. Pour le pays, corrigez l'emplacement de la clé primaire

Quelles sont les différences dès lors au niveau de l'utilisation de la fonction DLookUp ?

10.19 Fonctions Count et DCount (requête)

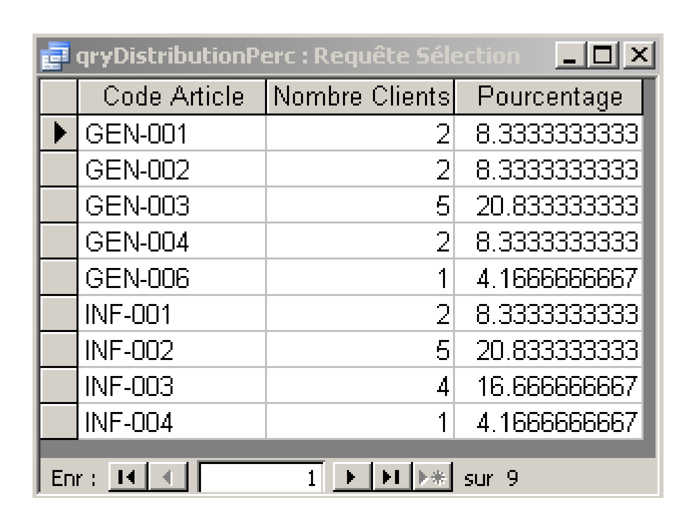
Nous continuons toujours avec notre table *tblSorties* et ce que nous désirons maintenant est connaître combien de clients ont commandé un certain produit (rien de nouveau) et savoir combien ils représentent en % du nombre total de clients ayant fait un achat.

La solution est loin d'être évidente (sans toutefois être extrêmement difficile). Nous donnons donc directement la solution:



La jointure s'expliquant par le fait que nous voulons quand même dans le listing les articles qui n'ont jamais été vendu!

Ce qui donnera:


qryDistributionPerc : Requête Sélection

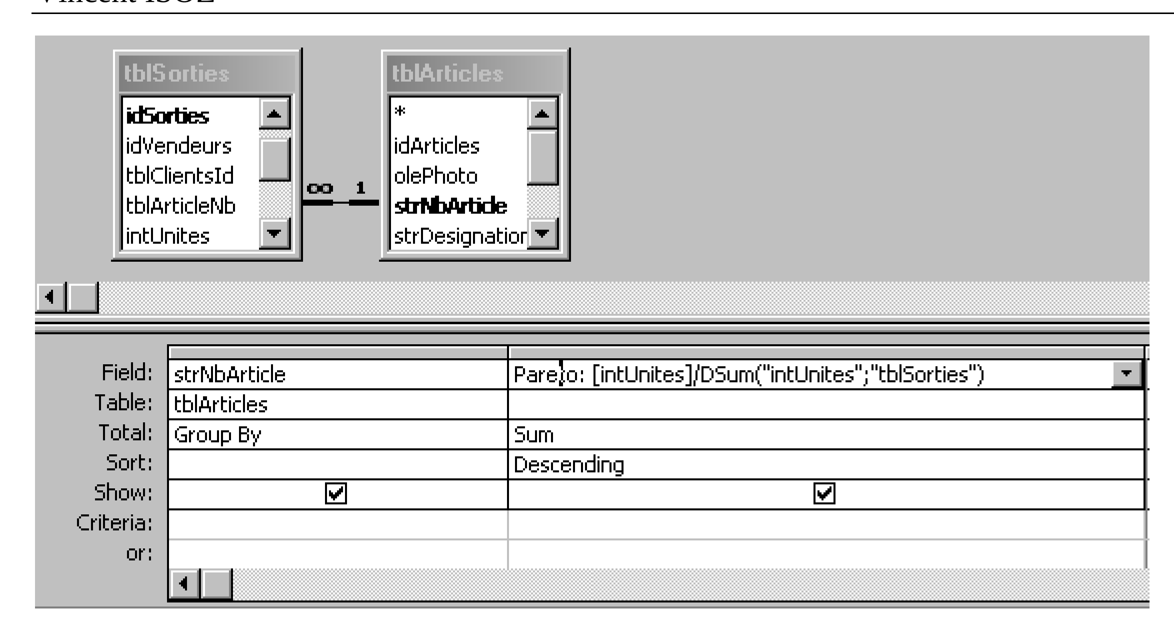
Code Article	Nombre Clients	Pourcentage
GEN-001	2	8.3333333333
GEN-002	2	8.3333333333
GEN-003	5	20.833333333
GEN-004	2	8.3333333333
GEN-006	1	4.1666666667
INF-001	2	8.3333333333
INF-002	5	20.833333333
INF-003	4	16.666666667
INF-004	1	4.1666666667

Enr : 1 sur 9

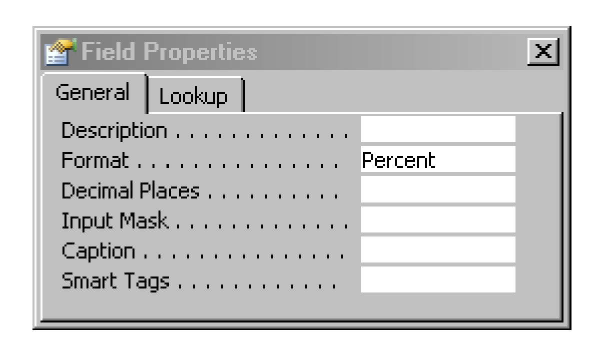
Requête que nous enregistrerons sous le nom *qryDistributionPerc*.

10.20 Fonction Dsum (requête)

Nous aimerions connaître la répartition des ventes par article. Il s'agit alors de faire une requête *qryPareto* avec:



en mettant la colonne *Pareto* au format %:



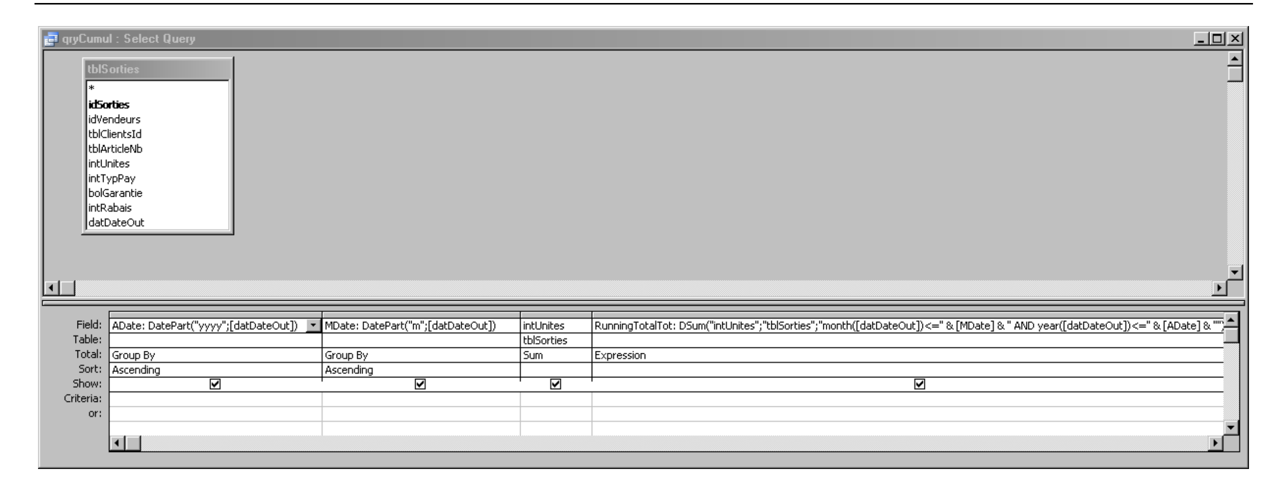
Ce qui donnera:

Code Article	Pareto
INF-002	51.30%
GEN-002	18.77%
INF-004	13.40%
GEN-003	6.08%
GEN-006	3.57%
GEN-001	3.57%
GEN-004	1.97%
INF-003	0.98%
INF-001	0.36%

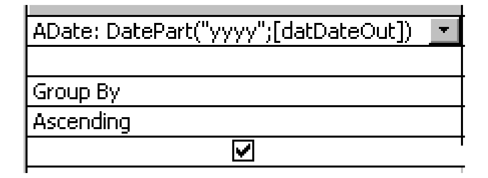
Il semblerait qu'en toute généralité, il ne soit pas possible de faire un vrai Pareto Cumulé sans du VBA dans MS Access.

10.21 Fonction Dsum et Dates (requête)

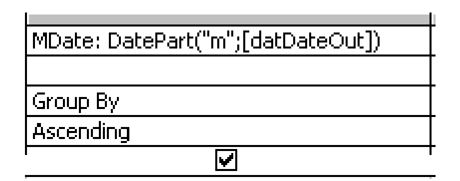
L'idée est de créer une requête qui cumula la vente des unités mensuelles avec un recommencement chaque année. Pour ce faire, la seule possibilité simple est la suivante:



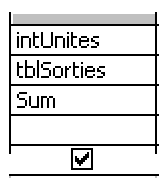
où nous avons dans la première colonne:



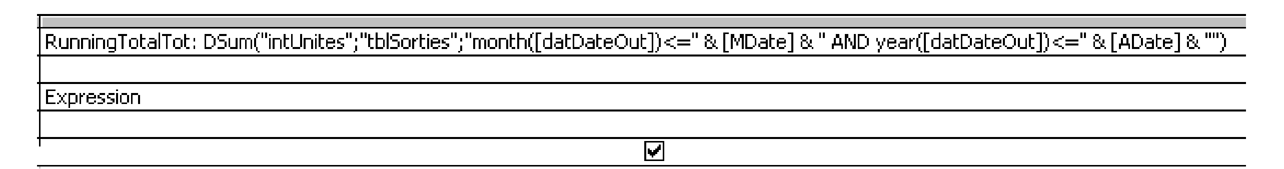
Dans la deuxième:



Dans la troisième:



Enfin dans la dernière:

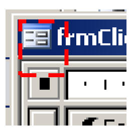


Attention!!! MS Access ne gère pas bien les fonctions en français. Il faut donc écrire tout en anglais afin que la requête ci-dessous fonctionne correctement!

11 Interfacage de l'application

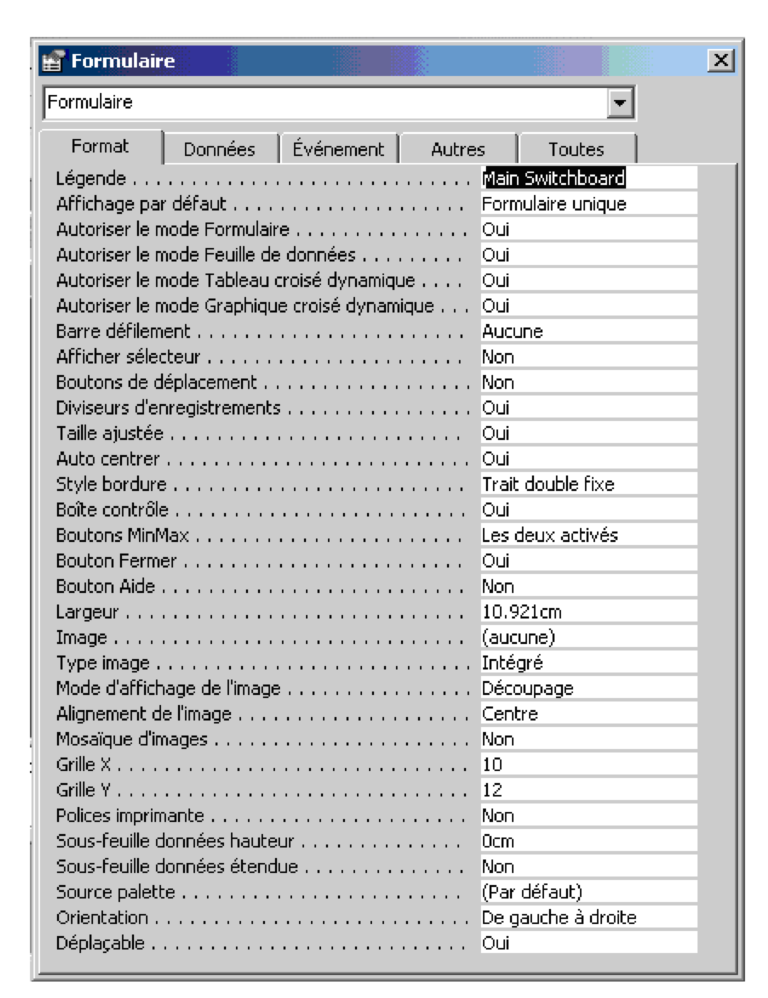
Les éléments principaux de la base de données ayant été vus, il convient maintenant de s'occuper de l'interfaçage global de l'application. Ce que nous entendons par là ? C'est l'interaction entre les formulaires, les requêtes, les sous-formulaires, les états, les événements, etc… et leurs propriétés avancées.

Avant de passer à des cas concrets il faudra se rappeler que pour chaque formulaire, vous le verrez (ou le savez déjà), qu'il y une petite case (voir figure ci-dessous) en mode création qui permet de paramétrer certaines options importantes du formulaire que vous êtes entrain de créer:



Parmi ces options, beaucoup ne sont utiles que pour la partie VBA ou macros du cours (dans l'onglet *Evenements*") donc nous ne nous y attarderons pas pour l'instant. Mais faisons un résumé de celles qui sont absolument indispensables avec un petit texte explicatif pour chacune d'entre elles.

11.1 Propriétés des formulaires



1. *Légende:* il faut toujours la saisir. C'est ce qui va apparaître à l'écran de vos utilisateurs. Souvent on y retrouve le nom de la société ou du logiciel c'est suivant…

2. *Barre de défilement*: c'est une option très importante. Beaucoup de formulaires n'en ont pas besoin donc inutile de perturber vos utilisateurs avec cela. De plus, quand c'est inutile ce n'est pas joli à l'écran et cela prend de la place.

3. *Afficher sélecteur:* cette option est inutile si votre base de données n'est pas en mode multiutilisateur sur un réseau. Effectivement, à quoi cela peut-il bien servir de voir un petit crayon à gauche du formulaire pendant que nous faisons de la saisie (à part de nous prendre pour des idiots…).

4. *Boutons de déplacement:* cette option vous permet de choisir si vous voulez les boutons de navigation (précédent, suivant, nouveau) en bas de vos formulaires. La plupart du temps il faut avouer que cette option est désactivée au profit de boutons qui seront plus esthétique et plus parlant pour les utilisateurs de votre application.

5. *Diviseur d'enregistrement*: vous pouvez la désactiver systématique mis à part (!) dans les sous-formulaires en mode tabulaire.

6. *Auto centrer:* toujours activer l'auto-centrage !! Cela vous évitera à vous énerver avec vos utilisateurs qui perdent le formulaire dans un coin de leur écran.

7. *Style bordure:* la plupart du temps, le développeur cherchera à mettre l'option *fine* correspondant à l'impossibilité pour l'utilisateur de changer la taille (par accident) de votre formulaire ce qui évitera des coups de téléphone au Help Desk pour rien…

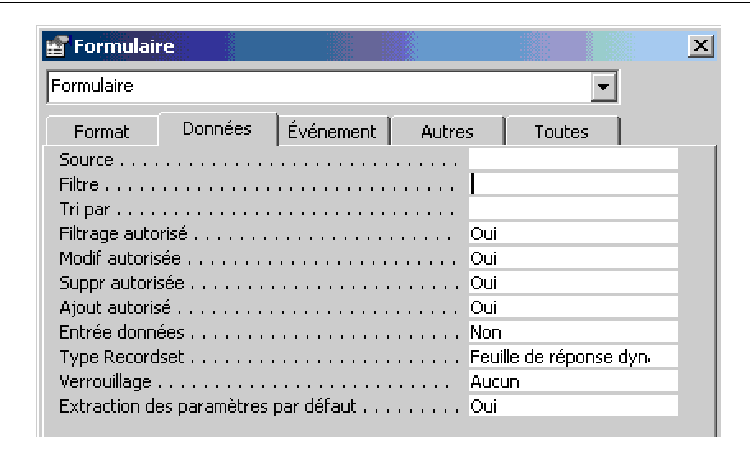
8. *Boîte de contrôle:* permet simplement de masquer l'affichage à l'écran sur le formulaire les trois boutons standards: *réduire, restaurer, fermer* (à vous de rajouter des boutons par la suite pour faire que l'utilisateur puisse quand même travailler…)

9. *Boutons min max*: permet simplement de masquer l'affichage à l'écran sur le formulaire les deux boutons standards: *réduire, restaurer, fermer*

10. *Bouton fermer:* désactive la possibilité d'utiliser le bouton *fermer* sans toutefois le masquer à l'écran.

11. *Déplacable:* empêche l'utilisateur de déplacer le formulaire à l'écran (ce qui est une très bonne chance…)

Toujours pour les formulaires voyons à présent ce qu'il y a d'intéressant dans l'onglet *Données:*



1. *Filtre:* vous pouvez saisir le nom d'un filtre que vous auriez enregistré à l'aide des filtres par formulaire (voir plus haut dans le présent support pour savoir faire cela)

2. *Tri par:* important mais pas dur à comprendre

3. *Filtrage autorisé:* à vous de voir si vous autorisez l'utilisation du filtrage

4. *Modif autorisée:* très utile ! (ne nécessite probablement pas d'explications)

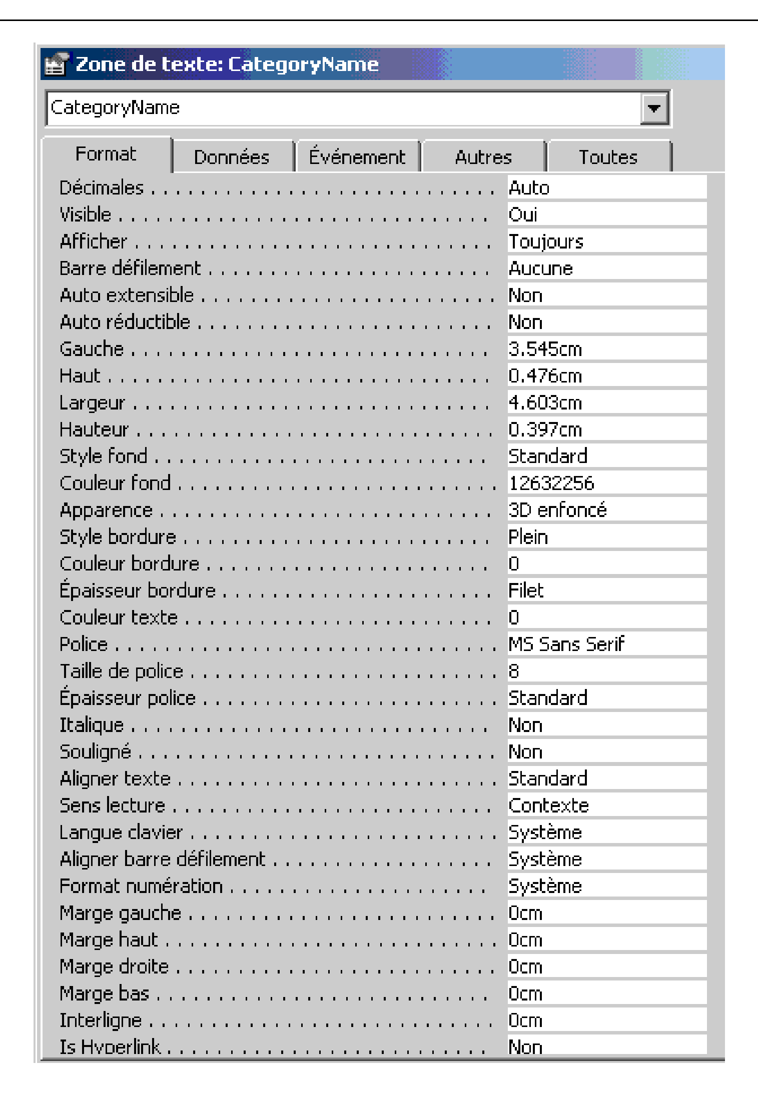
5. *Suppr autorisée:* très utile aussi ! (ne nécessite probablement pas d'explications)

6. *Ajout autorisé:* très utile aussi ! (ne nécessite probablement pas d'explications)

7. *Verrouillage:* très utile pour appliquer l'impossibilité de cliquer sur certains champs du formulaire actif.

11.2 Propriétés des champs de formulaires

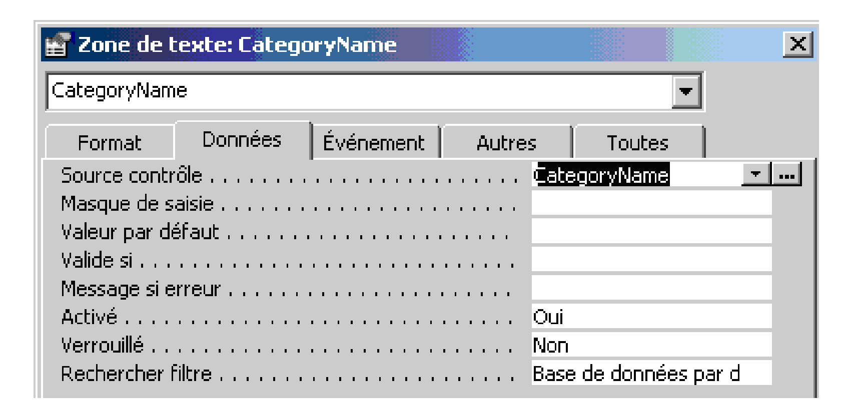
Les propriétés des champs de formulaires sont très nombreuses et beaucoup d'entre elles sont un peu inutiles techniquement parlant (mais pas graphiquement !!):



1. *Décimales*: nous avons déjà vu cela lors de notre travail dans les tables donc inutile de s'y attarder c'est trivial.

2. *Visible:* option très important permettant de masquer un champ à l'utiliser (typiquement pour faire des calculs intermédiaires indivisibles à l'utilisateur).

3. *Style fond:* il est parfois mis en mode transparent si nous souhaitons faire croire à l'utilisateur que l'information qui y est indiquée n'est pas dans un champ.

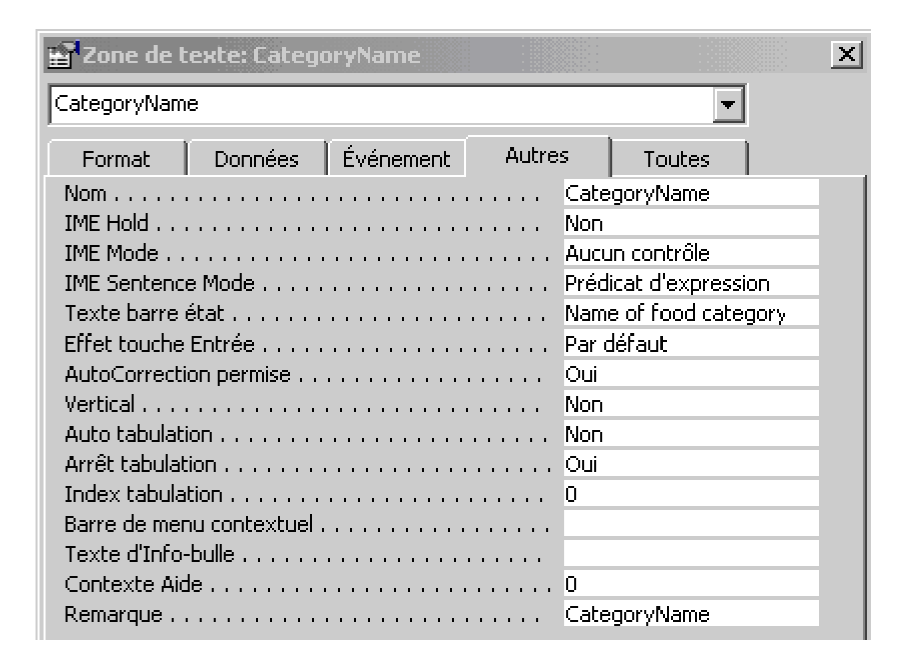


1. *Source contrôle*: permet de définir d'où viennent les données (si elles viennent de quelque part) pour le champ sélectionné

2. *Masque de saisie:* option très important car il ne faut pas oublier que les masques de saisie ne sont pas stockés dans les tables. Ainsi, dans un formulaire il peut arriver que vous ayez besoin de redéfinir un masque.

3. *Activé:* le champ devient grisé et l'utilisateur ne peut plus cliquer dessus mais seulement lire le contenu.

4. *Verrouillé*: l'utilisateur ne peut plus modifier le contenu mais peut cependant toujours cliquer dans le champ (contrairement à l'option *Activé*)



1. *Nom:* cette option est cruciale quand vous faites des calculs avec les champs. Nous l'avons déjà utilisée de nombreuses fois dans des exercices antérieurs.

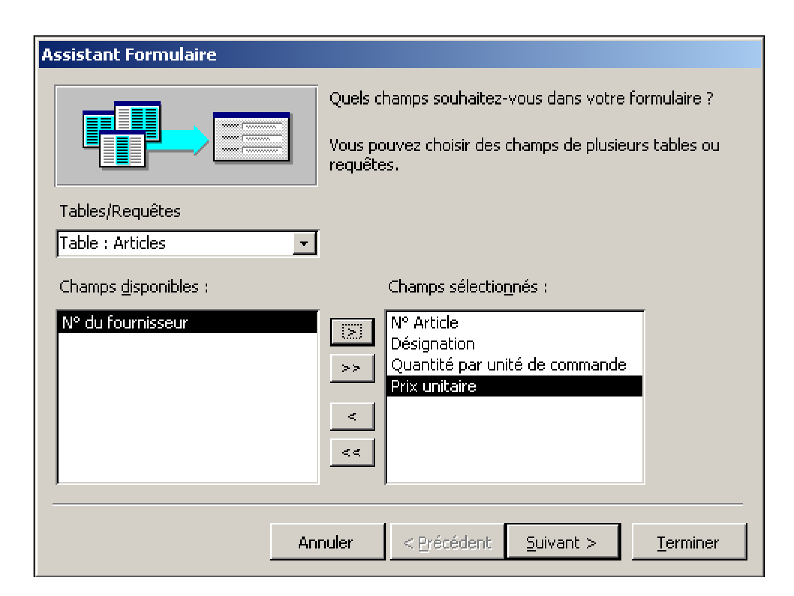
2. *Texte barre d'état:* petit truc sympa mais encore faudrait-il que les gens regardent en bas de leur écran….

3. *Texte d'infobulle*: souvent utilisé et important. les info-bulles sont connues par toutes et par tous.

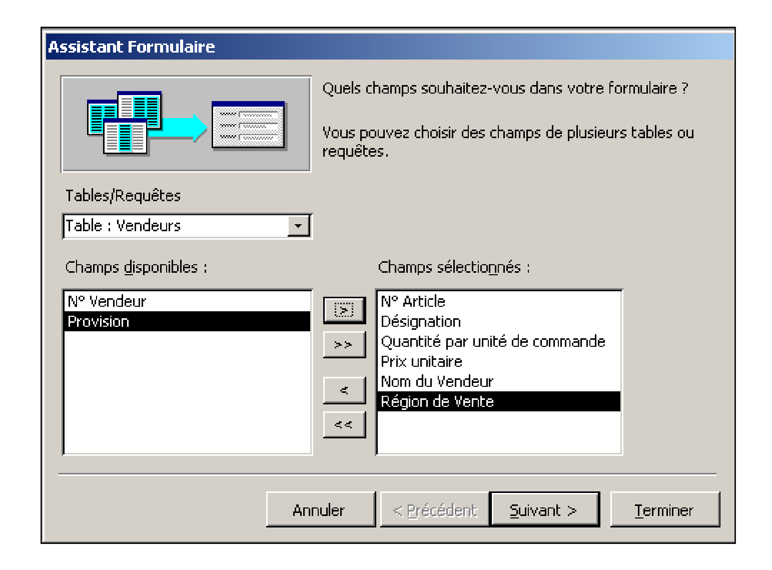
11.3 Sous-formulaires (Assistant)

Dans la base, créez à l'aide de l'assistant un formulaire contenant un sous-formulaire qui comporte les champs d'enregistrements des tables *tblArticles* et *tblVendeurs* sans redondance de champs.

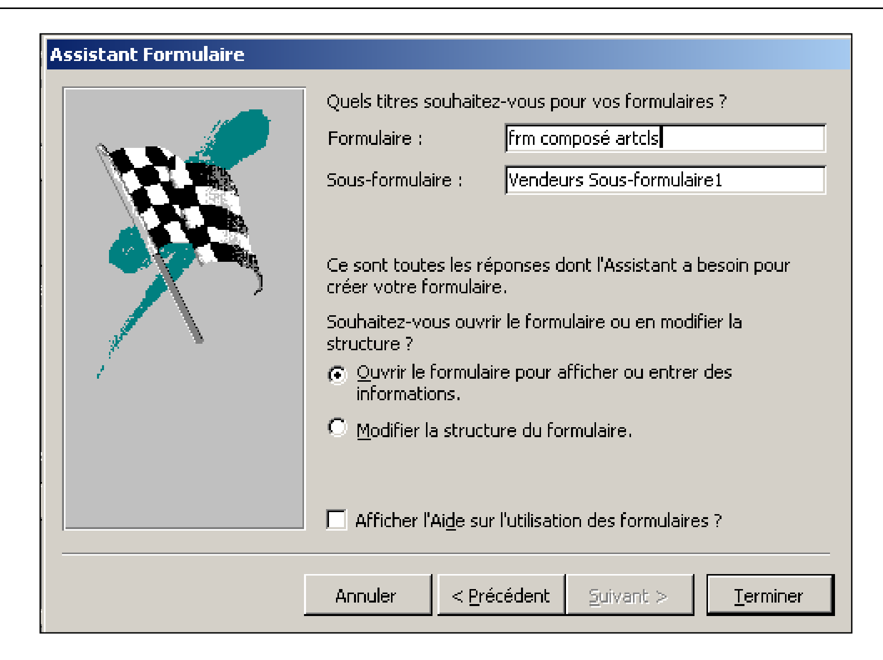
Le but étant de créer un formulaire qui indique pour chaque article, qui l'a vendu, où et en quelle quantité.



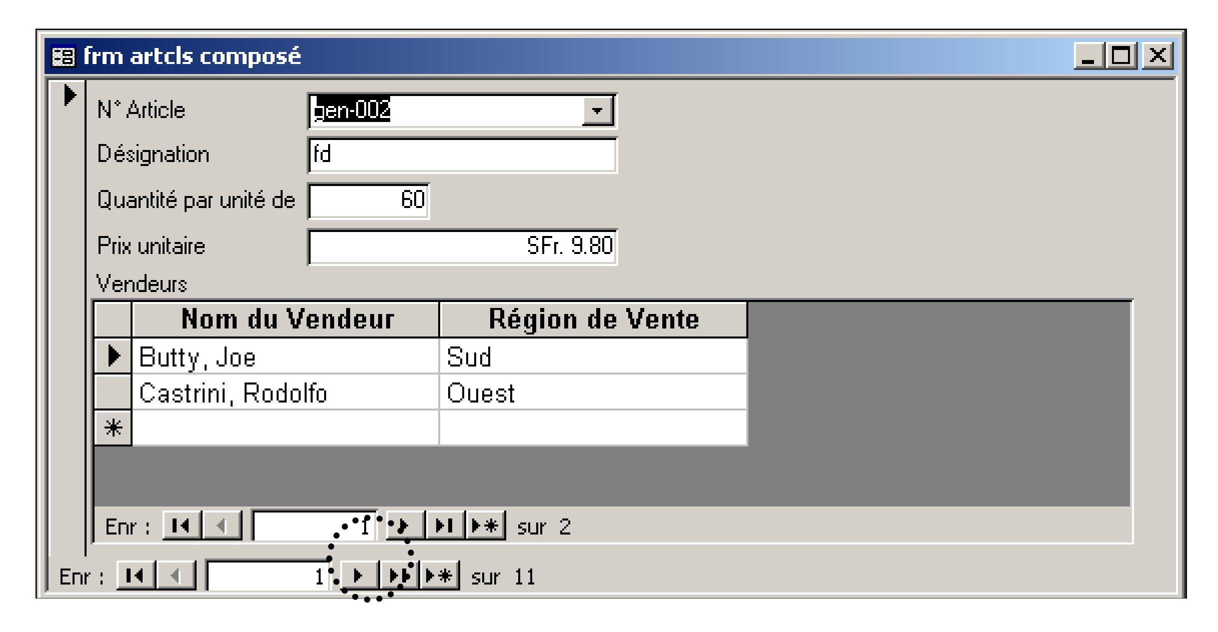
Ensuite avec la table *tblVendeurs*:



Valider 'Suivant >', choisir d'afficher les données par codes d'articles. Valider encore 'Suivant>' jusqu'à arriver à la boîte de dialogue ci-dessous et d'avoir les valeurs de champs correspondantes:



Parcourez ensuite les données en utilisant les flèches de navigation:



Il est possible de combiner plus qu'une table et ce qu'elle soit connexe ou non à la table principale du formulaire!

Attention!!! Le bouton *Dupliquer* de Microsoft Access ne fonctionne pas avec les sous-formulaires. Pour pouvoir faire l'équivalent de ce bouton avec un formulaire et un sous-formulaire il faut faire un code VBA (voir le chapitre correspondant).

11.4 Champs calculés

Dans le formulaire ci-dessus, créez en *mode création* un nouveau champ de données intitulé *Total*.

Affichez les propriétés de ce champ et dans l'onglet *Données* sur la ligne *Source de contrôle*, tapez le signe égal (=) puis précisez l'expression qui doit permettre à Access de calculer la valeur totale de la commande:

=[Quantité par unité de commande][Prix unitaire]*

N'oubliez pas le format du champ inséré qui doit être dans notre cas du type monétaire.

En plus, dans le même formulaire, créez un champ qui indique, lorsque le total est supérieur à 200.- francs, le fait que le vendeur ait eu le droit de faire une ristourne ou pas:

=IIF([Quantité par unité de commande][Prix unitaire]>200;"Ristourne";"Pas de Ristourne"]*

Un petit rappel sur les fonctions disponibles:

IIf(expression;résultat si vrai; résultat si faux): calcule l'expression en fonction de une ou plusieurs conditions.

Par exemple avec plusieurs conditions:

*IIf([Date commande]>#01/02/97# ET [MONTANT]>=10000;[Montant]*0.9;[Montant])*

Date() afficher la date du jour

DateDiff() calcule le nombre de jours entre deux dates

Month() extrait le mois d'une date donnée

Year() extrait l'année d'une date donnée

Sum() calcule la somme des valeurs d'un champ (ex. Sum([Prix unitaire]))

Avg() calcule la moyenne des valeurs

Log…, **sin**…

Pour concaténer deux champs, on utilisera la syntaxe:

[Titre] ***&*** *" "* ***&*** *![Nom]*

11.5 Sous-formulaires (Boîte à outils contrôle)

Ouvrez le formulaire article et agrandissez-le afin d'avoir une surface de travail disponible suffisamment grande pour y insérer un sous-formulaire.

Voici une autre méthode de création de sous-formulaires (et de sous-états !) qui est plus puissante que celle de l'assistant lorsque l'on a des formulaires qui ont des champs venant de tables multiples dont un de ceux-ci (ou plusieurs de ceux-ci) doivent contrôler le contenu du sous-formulaire (c'est un peu compliqué) ! Cette méthode permet également de créer autant de sous-formulaires que l'on désire !!

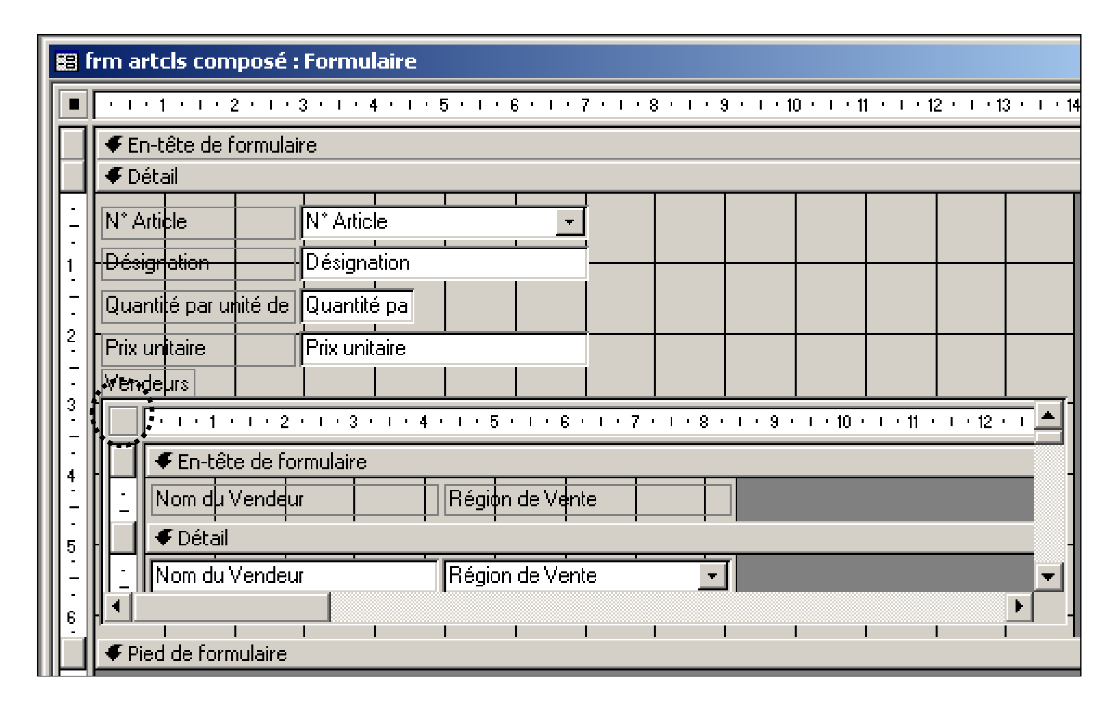
En mode création du formulaire articles cliquez sur le bouton "sous-formulaire" comme ci-dessous:

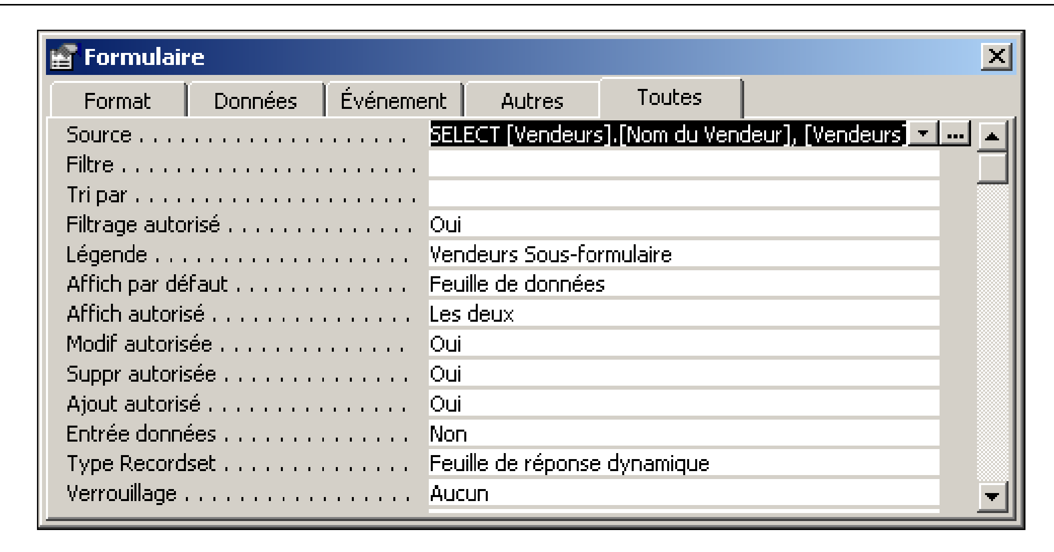


Ensuite apparaît un assistant à l'écran qui vous demande quelles données doit contenir le sous-formulaire. Sélectionnez la source depuis la table *tblSorties* (ou une formulaire de la table sorites que vous aurez créé au préalable) et ensuite prenez les champs de cette table excepté *N° Article* pour qu'il n'y ait pas redondance de l'information.

La question suivante de l'assistant qui vous demande de définir vous-même les champs à lier est particulièrement utile si vous avez un formulaire utilisant des champs de tables multiples dont un seul (ou plusieurs) doit contrôler le contenu du sous-formulaire (cela fonctionne un peu comme un filtre au fait…). En ce qui nous concerne ce n'est pas le cas et nous allons simplement cliquer sur le bouton "Suivant".

Vous pouvez définir des propriétés pour le sous-formulaire et les contrôles du formulaire. Ainsi en allant dans le mode création et en affichant l'onglet *Toutes* de la page de propriétés du sous-formulaire (clique gauche et *Propriétés*):





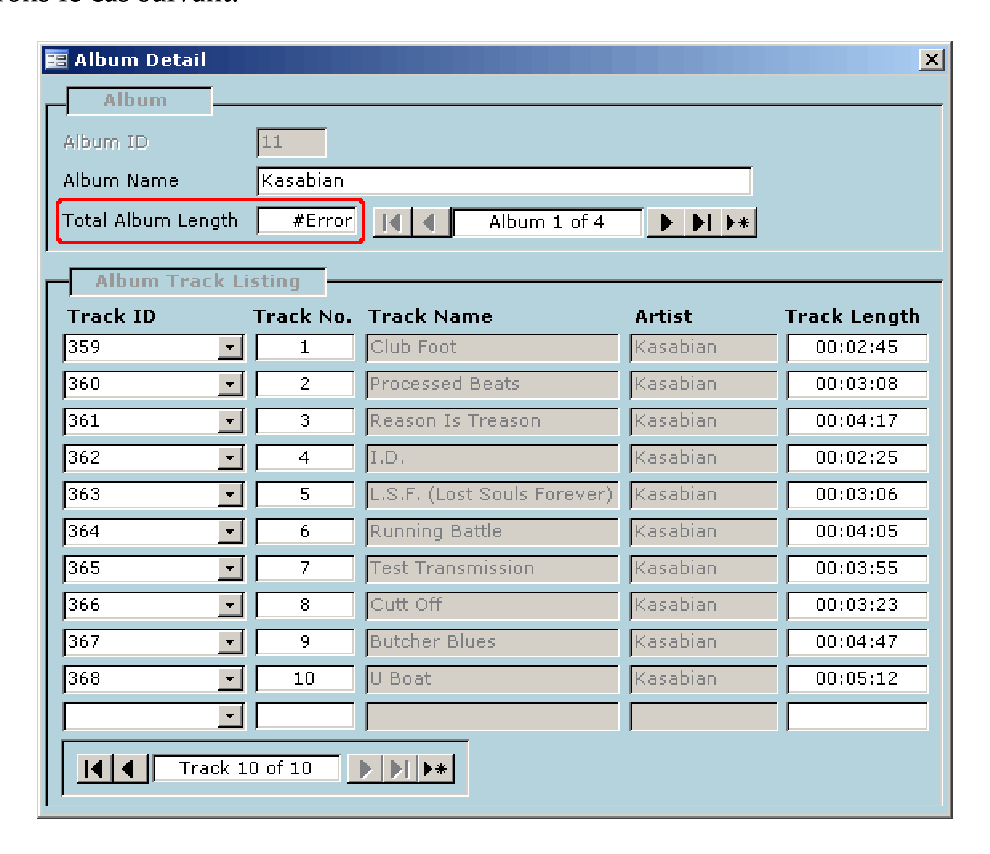
Vous avez ici de nombreuses options dont certaines sont forts intéressantes autant pour leur utilité que pour leur esthétique.

Sinon pour les éléments de formulaire:

L'option *Activé* (oui/non) pour l'accès à un champ et *Verrouillée* (oui/non) pour la modification d'un champ sont à peu près équivalentes (validez et testez!).

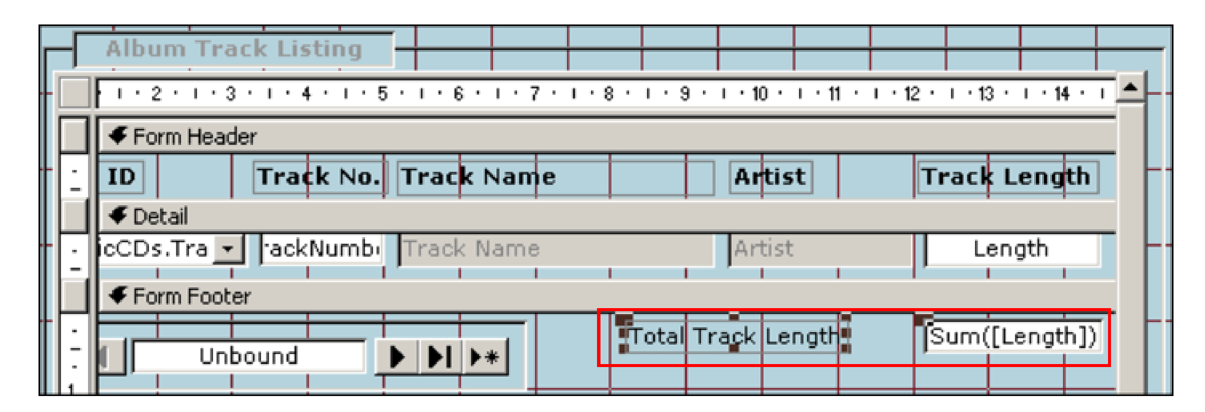
11.5.1 Total d'un sous-formulaire dans le formulaire principal

Considérons le cas suivant:



Nous avons une erreur car les temps des pistes de notre CD sont à l'intérieur du sous-formulaire. Dès lors, il faut savoir que pour arriver à reporter le total dans le formulaire

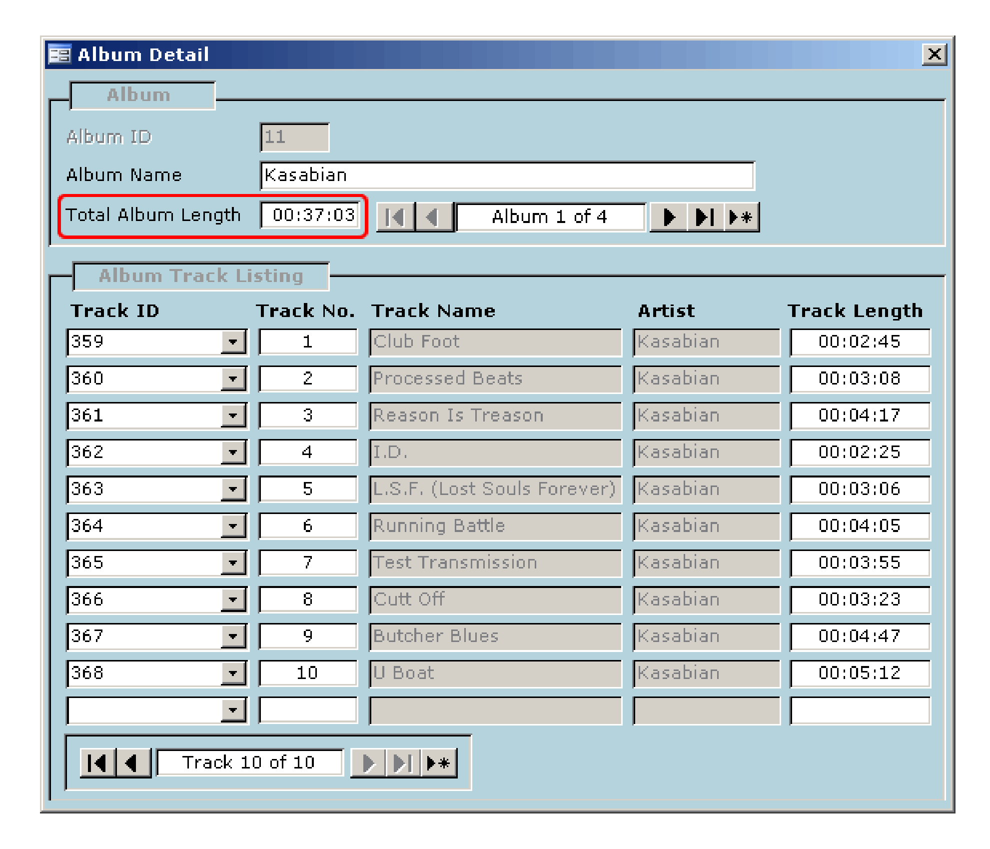
principal il faut d'abord faire la somme dans le sous-formulaire dans le pied ou l'en-tête de ce dernier (idéalement on le fera dans le pied):



Et enfin, dans le formulaire principal, on peut écrire dans le champ:

=[sfrmAlbumTracks].[txtTotalTrackLengths]

Ce qui donnera:

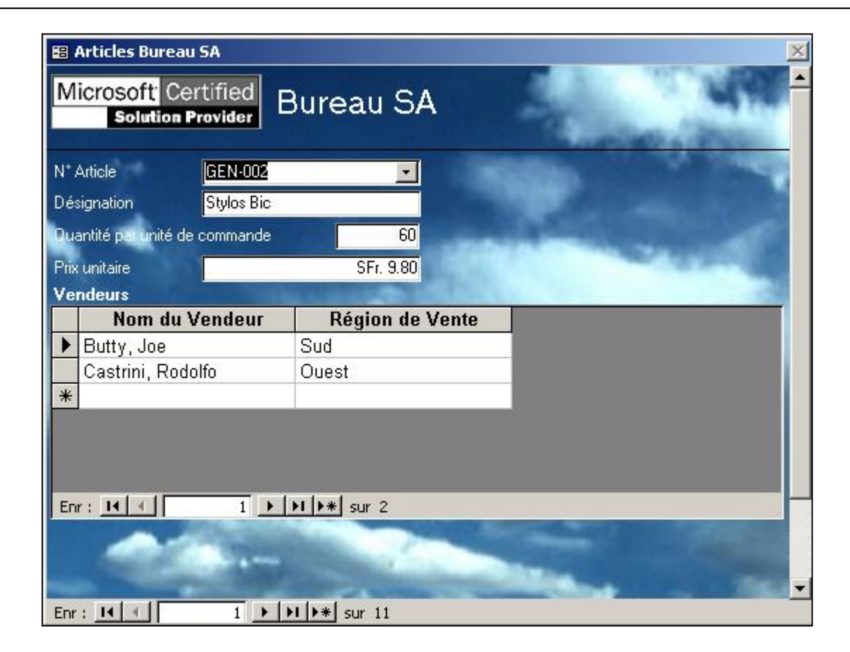


11.6 Formatage

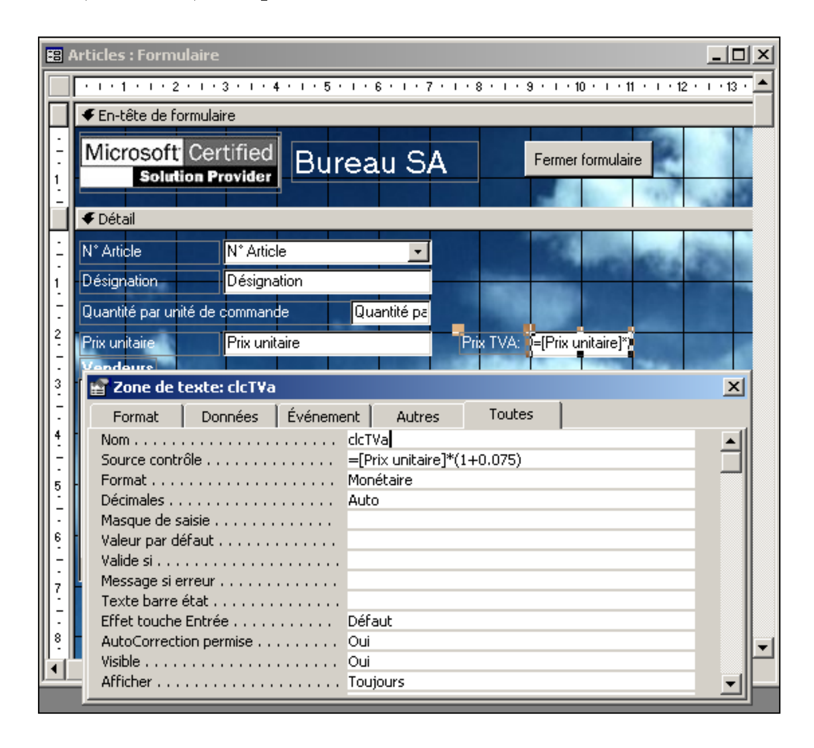
Ouvrez cette dernière base et le formulaire *frmComposeArticles* en mode création et changez le formulaire afin d'obtenir le résultat suivant ci-dessous:

L'image de fond a été définie dans les propriétés du formulaire dans l'option *Images*.

Le logo quant à lui se place avec les options disponibles dans la barre d'outils.



Créez dans le formulaire ci-dessus, un "contrôle calculé" nommé *clcTVA* et dont la formule sera =[Prix unitaire]*(1+0.075) tel que:



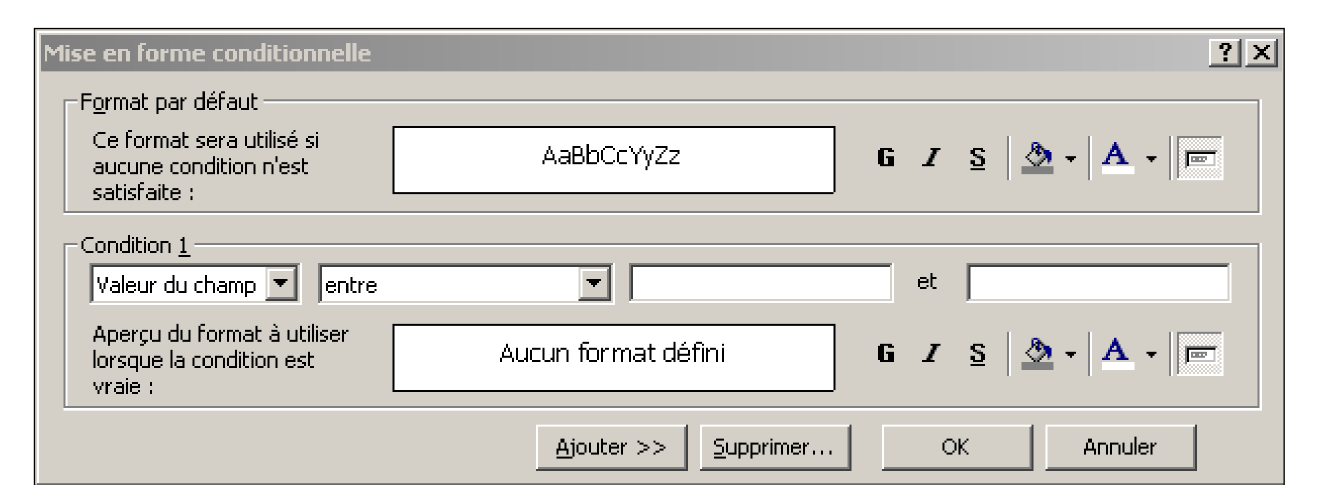
Si vous regardez la taille de la base maintenant vous verrez que sa taille tourne autour des 10 MB !!! Au fait, MS Access convertit l'image que l'on a utilisée en arrière-plan en image bitmap… On verra plus tard comment l'on peut réduire la taille de la base de façon relativement satisfaisante.

Maintenant, ajoutez un bouton "Fermer le formulaire" qui permette à l'utilisateur de fermer le formulaire de façon conviviale et ergonomique.

Sachez que vous pouvez également formater le contenu d'un contrôle avec la barre d'outils mis en forme:



Vous avez également une option très intéressante qui comme dans Excel permet de définir une mise en forme conditionnelle pour un champ de données:



Ces différentes options étant extrêmement simple d'utilisation, je n'ai pas souhaité créer d'exercices écrit les utilisant. Amusons-nous un peu avec les formulaires que nous avons actuellement à disposition dans note base.

Remarque: attention on ne peut pas désactiver un champ de formulaire (clic droit/propriétés/données/activer) qui a un format conditionnel !!

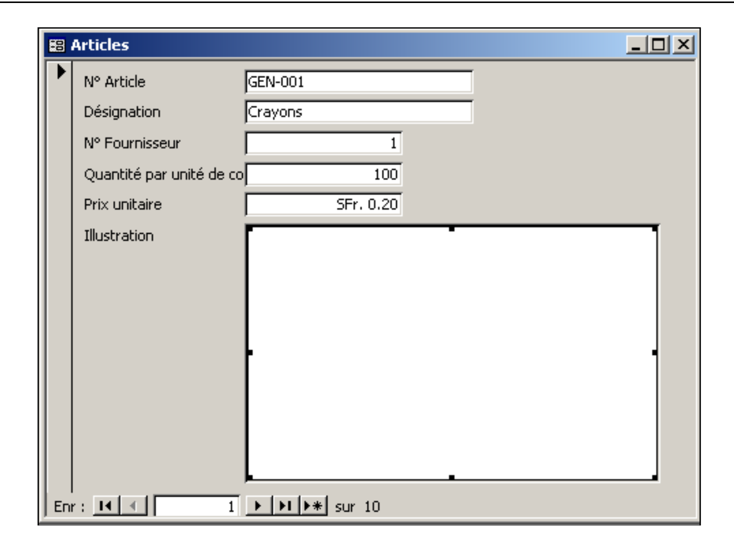
11.7 Champs OLE

Il est fréquent que dans certains cas particuliers de bases de données, on ait besoin d'intégrer une recherche d'archives de documents. Or, pour arriver à intégrer des objets dans une base Access (opération fonctionnelle mais sensible du point de vue de la stabilité de la base) il est bien évidemment nécessaire de définir un champ de table bien spécifique.

Pour essayer cette fonctionnalité, je vous propose de rouvrir la base Magasin de d'ouvrir la table nommée *Articles* en mode création et d'y insérer un champ nommé *Illustration* de type Objet OLE.

Fermez et enregistrez ensuite la table et créez à partir de cette dernière un formulaire instantané. Vous devriez dès lors obtenir un formulaire comme celui représenté sur la capture d'écran visible à la page suivante.

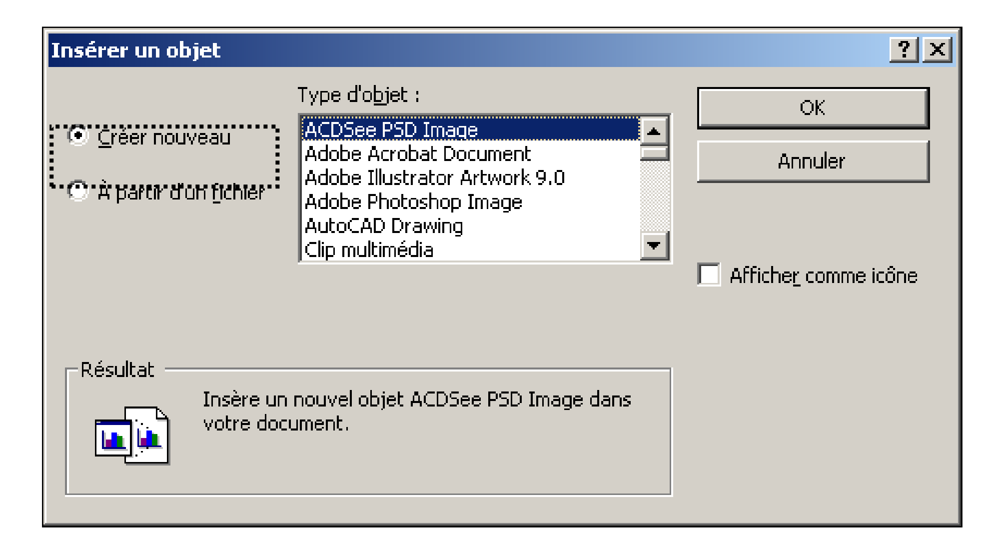
Nous ne reviendrons pas sur la méthode de personnalisation de formulaire, chose qui devrait être bien maîtrisée à ce niveau du cours.



Que faire de cette grosse zone blanche dans le formulaire ? En fait, elle représentera une image (sous forme d'icône si on le désire) de l'objet lié à l'enregistrement en cours d'affichage.

Exemple:

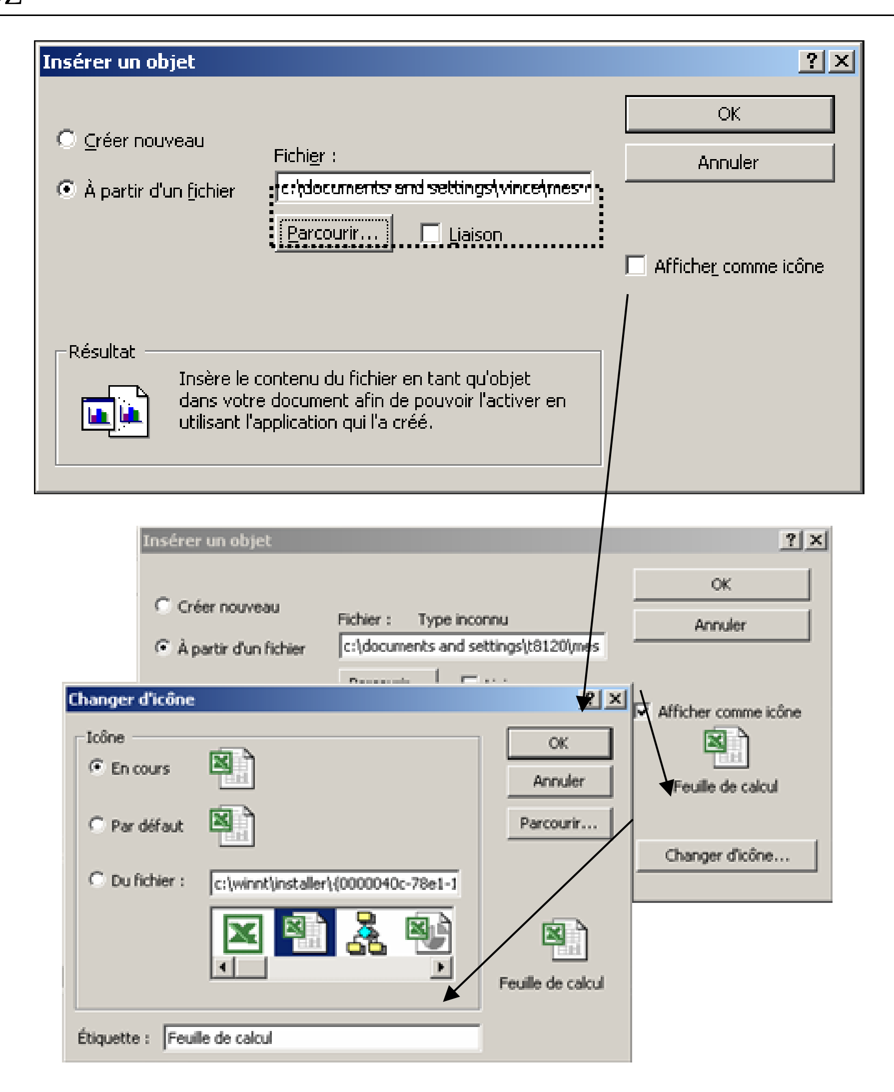
Pour lier une image à l'article GEN-001, il faut aller dans le menu *Insertion/Objet*. Ensuite, apparaît la boîte de dialogue ci-dessous qui devrait (au niveau d'un créateur de bases de données Access déjà connue):



Déjà on se voit confronté à un problème au niveau de l'utilisation des champs OLE. Leur utilisation n'est pas triviale et l'utilisateur lambda n'aura probablement jamais l'idée d'aller intuitivement dans le menu *Insertion*. Nous verrons plus tard, si nous pourrons rendre cette action un peu plus "user-friendly".

La première option nommé "Créer nouveau" est quasiment jamais utilisée. Nous ne nous y attarderons donc pas.

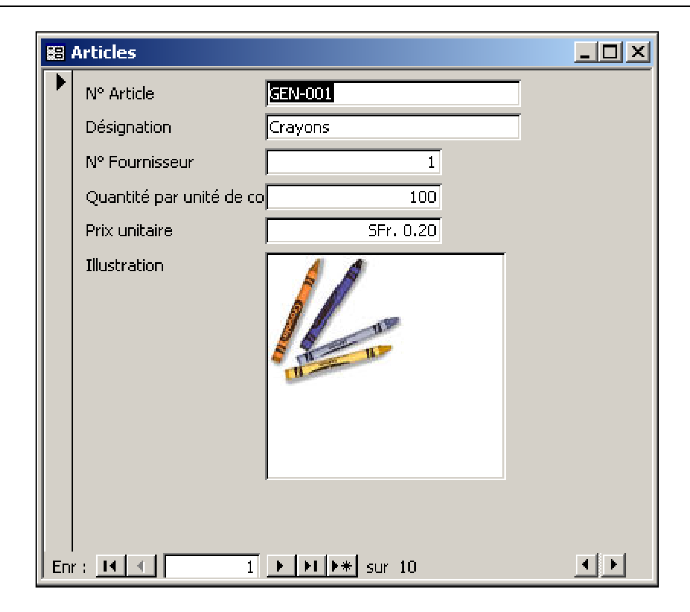
Sélection donc l'option "A partir d'un fichier" et cliquer ensuite sur le bouton "Parcourir" qui devrait être accessible dès lors à l'écran (comme représenté à la page suivante).



Il est de première importance de remarquer l'option nommée "Liaison" qui vous est proposée. Son but, rappelons-le quand même au cas où, est de permettre de modifier le fichier OLE (Object Linked and Embedded) sans l'ouvrir depuis MSAccess et que celui-ci fasse automatiquement une mise à jour de la liaison (au fait, c'est bien plus compliqué que cela mais nous n'avons pas le temps d'aller plus dans les détails).

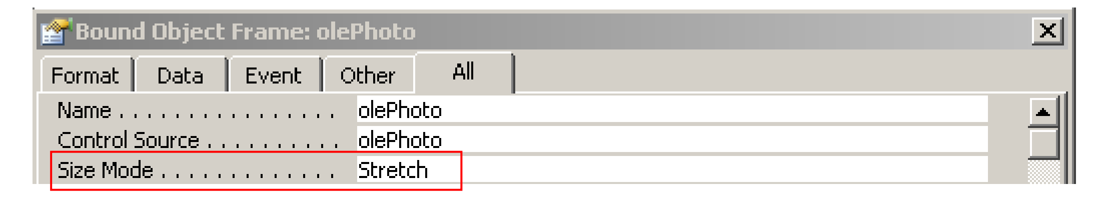
Ce type d'option est certainement fort intéressant mais demande, lors d'une distribution en réseau de la base, d'avoir une bande passant non négligeable, lorsque ce genre d'opérations (ouverture de fichiers liés) sont fréquemment effectuées par les utilisateurs. De plus, les documents liés devraient éviter d'avoir trop la "bougeotte" sur l'arborescence du réseau de l'entreprise. En plus, bien que le fichier ne soit que lié, celui-ci est quand même intégré au fichier *.mdb dont je vous laisse imaginer le poids mémoire lors d'une utilisation accrue de ce genre d'options.

Allez chercher le fichier nommée "crayons.jpg" (attention à la valeur du champ "Nom" de la boîte de dialogue "Parcourir" qui peut parfois vous causer des surprises). Après avoir trouvé ce fichier image et validé la chose, vous devriez avoir un formulaire ressemblant à la capture d'écran ci-dessous:

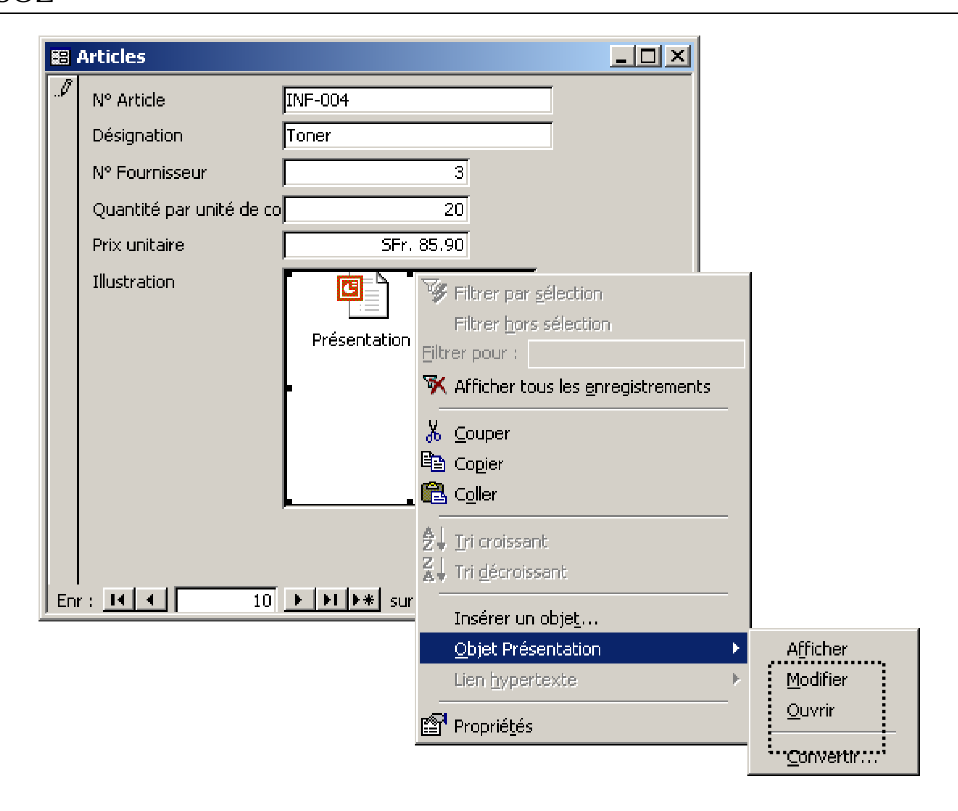


Répétez l'opération avec tous les autres articles et testez si le système fonctionne bien (de plus c'est un très bon exercice). Enregistrez les modifications de votre formulaire une fois terminé (en prévision de la suite…).

Remarque: si vous souhaitez que l'image s'adapte automatiquement à la taille du champ, faites un clic droit sur celui-ci et choisissez l'option *Propriétés*. Ensuite allez dans l'onglet *Tous* et dans la propriété *Mode d'affichage* sélectionnez *Echelle*:



Si l'objet OLE inséré est un document de la suite Office éditable, l'utilisateur de la base de données n'a (théoriquement) qu'à double cliquer sur l'icône du document pour pouvoir l'éditer. Malheureusement, cela ne fonctionne pas toujours. Il faut alors que l'utilisateur clique avec le bouton droit de la souris dessus de façon à avoir accès aux options suivantes qui elles sont quasiment toujours fonctionnelles:



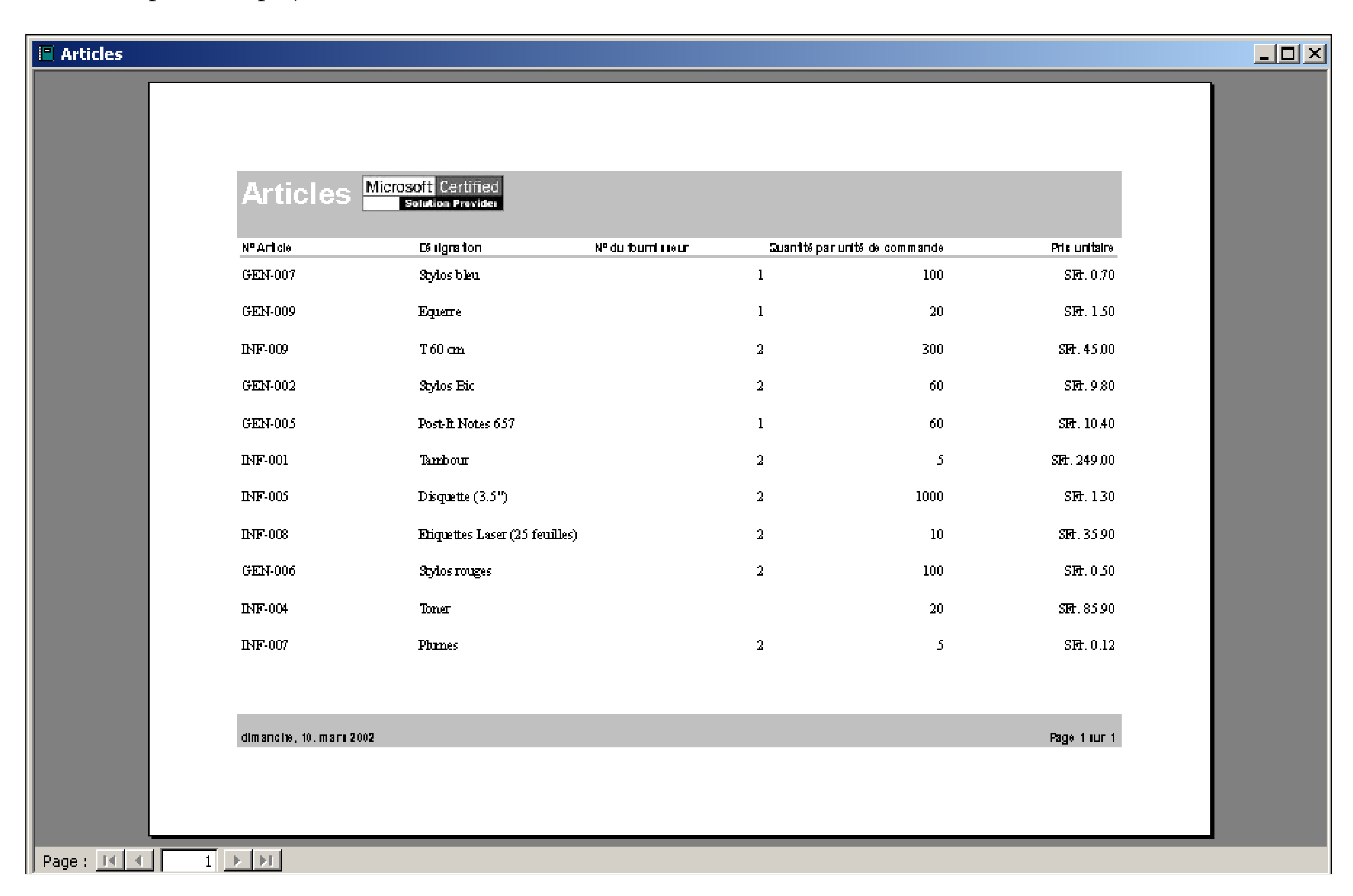
Je vous laisse le temps d'essayer cette option avec un fichier texte quelconque que vous prendrez le soin de préparer et d'insérer par vos propres moyens et connaissances.

Prenez également le temps de tester l'option "Liaison" !

Remarque: il est possible que MS Access rencontre parfois un problème avec le serveur OLE. Au fait, c'est MS Windows qui a le problème et non MS Access… pour palier à ceci, vous pouvez dès lors insérer les objet par copier/coller ou aller corriger l'erreur dans la base de registres de MS Windows mais alors là… nous sortons du cadre de cette formation.

11.8 Esthétique rapports

Sélectionnez la table *tblArticles* et créez-en un état instantané (comme celui ci-dessous) avec une image personnalisée (logo de votre entreprise enregistrée depuis l'Internet par exemple):

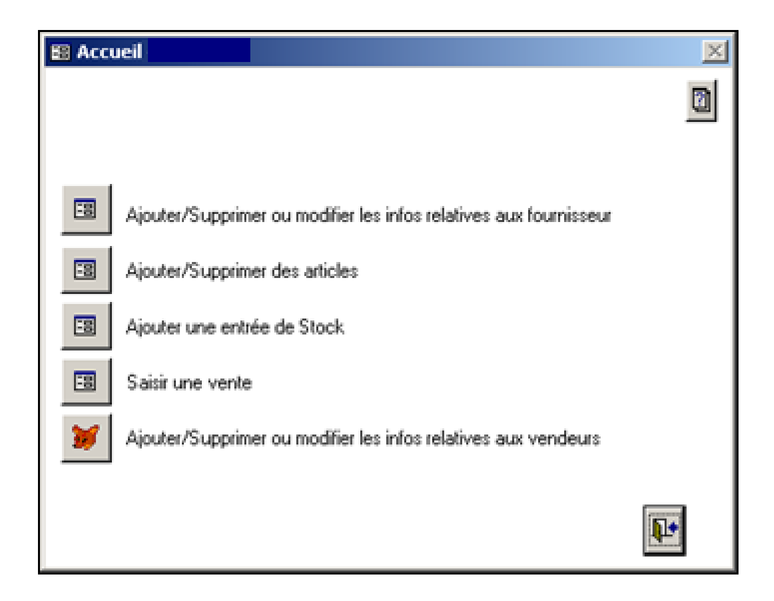


N° Article	Désignation	N° du fournisseur	Quantité par unité de commande	Prix unitaire
GEN-007	Stylos bleu	1	100	SFr. 0.70
GEN-009	Equerre	1	20	SFr. 1.50
INF-009	T 60 cm	2	300	SFr. 45.00
GEN-002	Stylos Bic	2	60	SFr. 9.80
GEN-005	Post-It Notes 657	1	60	SFr. 10.40
INF-001	Tambour	2	5	SFr. 249.00
INF-005	Disquette (3.5")	2	1000	SFr. 1.30
INF-008	Etiquettes Laser (25 feuilles)	2	10	SFr. 35.90
GEN-006	Stylos rouges	2	100	SFr. 0.50
INF-004	Toner		20	SFr. 85.90
INF-007	Plumes	2	5	SFr. 0.12

dimanche, 10. mars 2002 — Page 1 sur 1

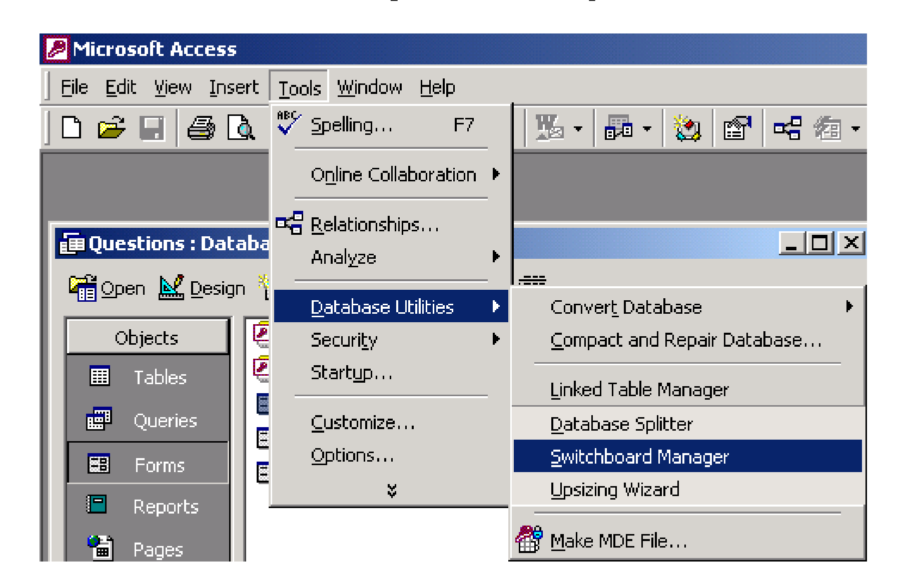
11.9 Formulaire de démarrage (switchboard)

Créez une nouveau formulaire vierge en mode création (théoriquement il est déjà terminé) en insérez des boutons (à partir le barre d'outils de contrôles) permettant d'effectuer toutes les opérations nécessaires à une bonne utilisation de notre base de données (le formulaire devrait quand même être bien mieux que celui présenté sur la figure ci-dessous…):

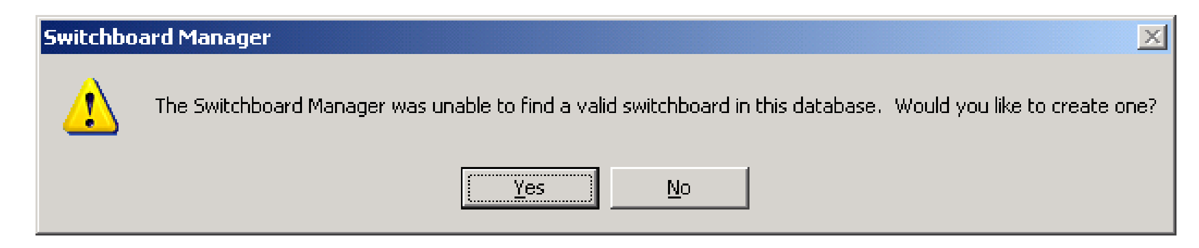


Il existe dans MS Access une sorte d'assistant pour faire un formulaire de démarrage (mais faut aimer…), appelé "Switchboard" en anglais.

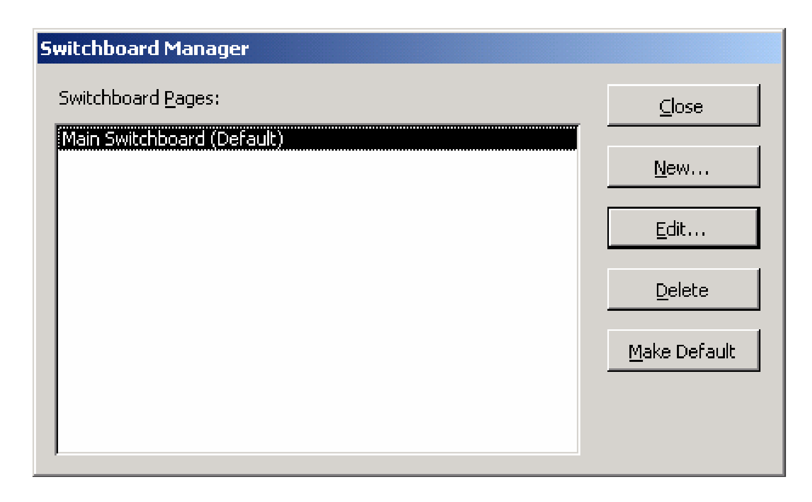
Pour lancer cet utilitaire il faut aller à l'emplacement indique ci-dessous:



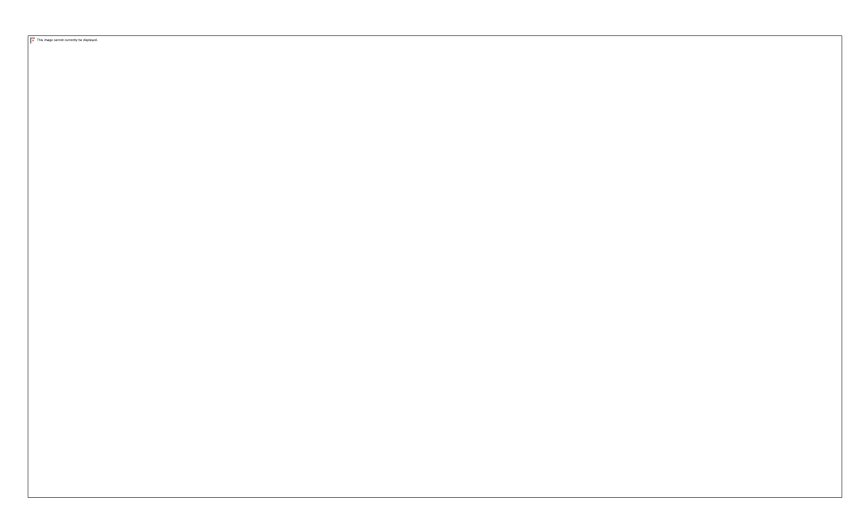
Apparaît ensuite une boîte de dialogue vous disant qu'il n'existe pour ce moment pas de formulaire de démarrage dans votre base. Cliquez sur *Yes* pour confirmer que vous souhaitez en créer un.



Quand l'assistant *Switchboard Manager* apparaît, cliquez sur le bouton *Edit:*



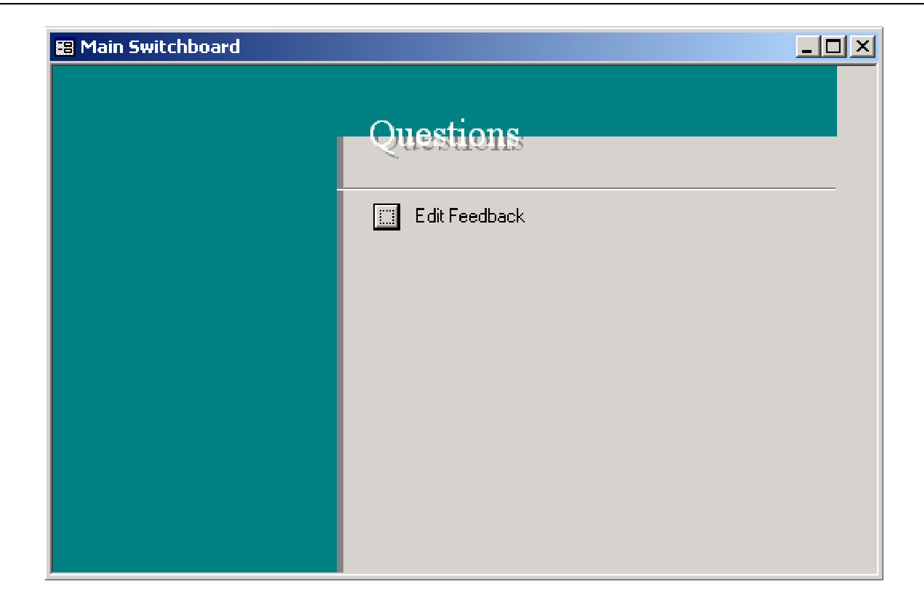
Ensuite, vous allez devoir ajouter des boutons à votre formulaire de démarrage. Pour faire ceci, cliquez sur le bouton *New*...



Entrez le texte qui va apparaître à côté du bouton. Dans cet exemple, nous création un élément de switchboard pour ouvrir un formulaire nommé *frmFeedback*. Cliquez sur le bouton *OK*.

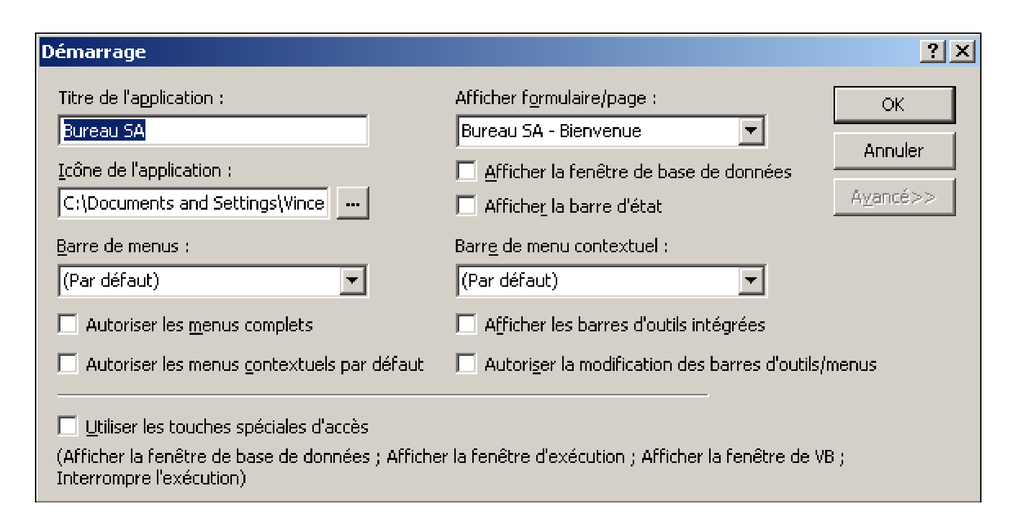


Maintenant, quand vous ouvrez votre base de données, le switchboard devrait s'ouvrir automatique et afficher les boutons que vous avez configuré:



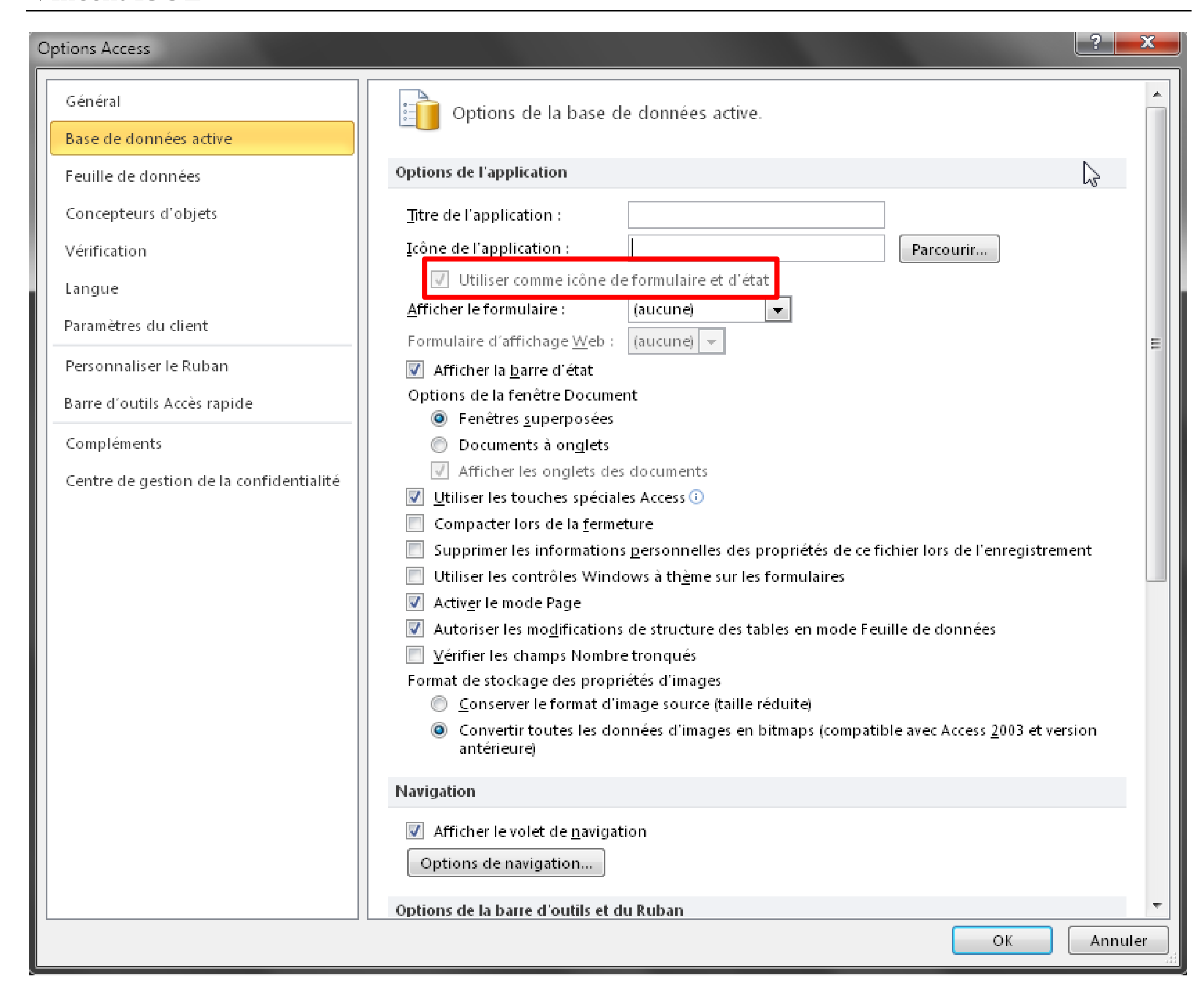
11.10 Options de démarrage

Définissez les paramètres nécessaires afin que le formulaire de Bienvenue démarre automatiquement (touche Shift pour annuler l'AutoExec[7]…) selon:



Cette boîte de dialogue a beaucoup changé et propose une nouvelle option mis en évidence sur la capture suivante depuis la version 2007:

[7] Il faudra passer par le système de gestion de sécurité de MS Access ou par une MDE ou par un code VBA pour bloquer la touche Shift (tous ces sujets sont traités plus loin)



Activez ou désactivez la case à cocher *Utiliser les touches spéciales* d'accès pour activer ou désactiver les touches suivantes:

Touches	**Effet**
F11 ou ALT+F1	Place la fenêtre Base de données au premier plan
CTRL+G	Appelle la fenêtre d'exécution
CTRL+F11	Bascule entre la barre de menus personnalisée et la barre de menus intégrés
CTRL+PAUSE	Dans un projet Microsoft Access, empêche Access de récupérer des enregistrements sur le serveur.

Tableau 5 Raccourcis clavier démarrage

Donc évidemment, si nous désactivons toutes les possibilités de l'utilisateur il y a un problème.... Il connaîtra très probablement la technique pour passer outre le blocage en appuyant sur la touche Shift du clavier pendant l'ouverture de la base et pourra accéder à toute sa structure. Il existe alors deux façons de faire...

1. Utiliser le système de gestion de la sécurité avacanée disponible dans les vieux fichiers *.mdb (système de sécurité qui n'est donc plus disponible lorsque l'on travaille avec les nouveaux formats de base de données Access 2007 et ultérieur)

2. Bricoler des codes VBA comme nous allons le voir ci-après car c'est le cas le plus demandé par les utilisateurs.

Microsoft propose alors le script assez simple suivant qu'il suffit dans un premier temps de mettre dans un module quelconque du projet VBA de votre base de données (projet qui sera évidemment protégé par un mot de passe!!!):

```
Function ap_DisableShift()
'This function disable the shift at startup. This action causes
'the Autoexec macro and Startup properties to always be executed.

On Error GoTo errDisableShift

Dim db As DAO.Database
Dim prop As DAO.Property
Const conPropNotFound = 3270

Set db = CurrentDb()

'This next line disables the shift key on startup.
db.Properties("AllowByPassKey") = False

'The function is successful.
Exit Function

errDisableShift:
'The first part of this error routine creates the "AllowByPassKey
'property if it does not exist.
If Err = conPropNotFound Then
  Set prop = db.CreateProperty("AllowByPassKey", dbBoolean, False)
  db.Properties.Append prop
  Resume Next
Else
  MsgBox "Function 'ap_DisableShift' did not complete successfully."
  Exit Function
End If

End Function

Function ap_EnableShift()
'This function enables the SHIFT key at startup. This action causes
'the Autoexec macro and the Startup properties to be bypassed
'if the user holds down the SHIFT key when the user opens the database.

On Error GoTo errEnableShift

Dim db As DAO.Database
```

```
Dim prop As DAO.Property
Const conPropNotFound = 3270

Set db = CurrentDb()

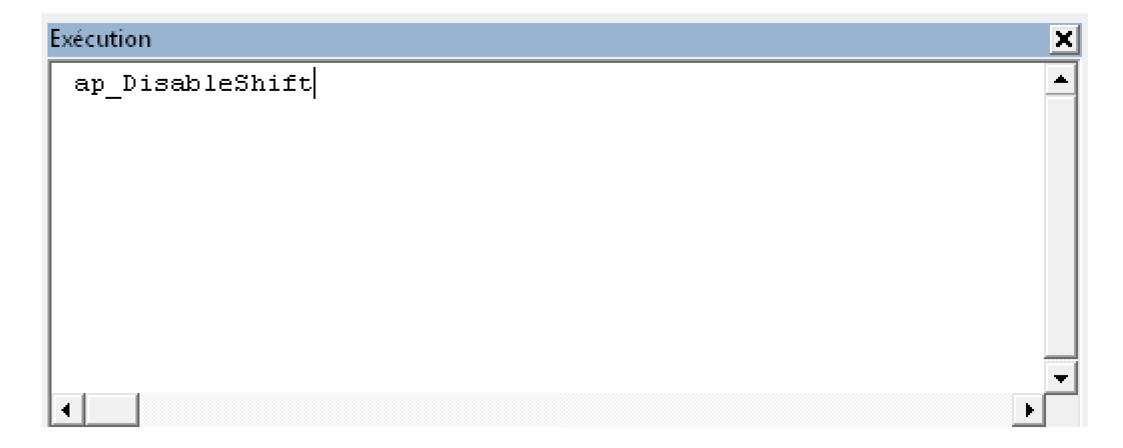
'This next line of code disables the SHIFT key on startup.
db.Properties("AllowByPassKey") = True

'function successful
Exit Function

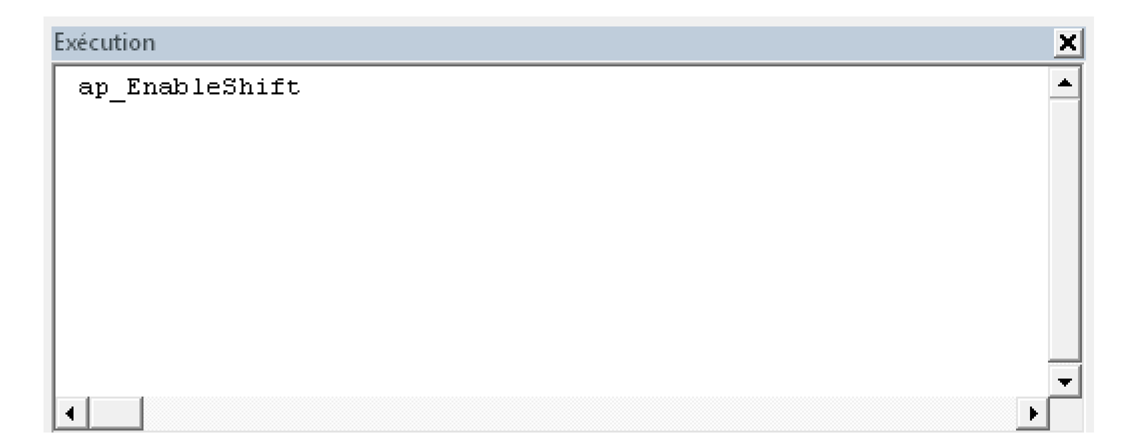
errEnableShift:
'The first part of this error routine creates the "AllowByPassKey
'property if it does not exist.
If Err = conPropNotFound Then
  Set prop = db.CreateProperty("AllowByPassKey", dbBoolean, True)
  db.Properties.Append prop
  Resume Next
Else
  MsgBox "Function 'ap_DisableShift' did not complete successfully."
  Exit Function
End If

End Function
```

Et ensuite dans la fenêtre d'exécution il suffira de taper respectivement pour activer le blocage:



et pour réactiver son utilisation:



Cependant, même si maintenant l'utilisateur ne peut plus faire Shift pour désactiver l'Autoexec, il lui reste la possibilité de faire F11 pour réactiver la fenêtre de base de données sous l'hypothèse que nous n'avons pas désactivé (il ne faut pas!!!) l'option:

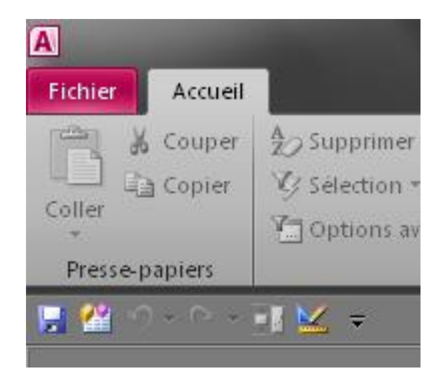


Remarque: Le Alt+F11 pour accéder au VBA n'est pas un problème puisque celui-ci est normalement protégé par un mot de passe et puis pour le problème du Ctrl+G qui permet d'accéder à la fenêtre d'exécution... eh bien il suffit simplement de nommer les deux fonctions de Microsoft autrement qu'avec leur nom par défaut sinon bien évidemment tous les petits curieux pourront désactiver votre pseudo-sécurité.

L'idée est alors (outre le fait de renommer les deux fonctions proposées par Microsoft) de mettre dans le formulaire de démarrage et en toute généralité dans tous vos formulaires (pas d'autre choix! car cela ne fonctionne pas si vous le mettez dans une routine centralisée) un appel à une routine via l'événement:

```
Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
   If KeyCode = vbKeyF11 Then KeyCode = 0
    MsgBox "Bien essayé!!!...."
End Sub
```

À partir de là votre base est relativement bien protégée. Cependant le malin qui personnalisera sa barre d'accès rapide comme il convient (c'est-à-dire avec le bouton *Mode création* ou *Éditeur Visual Basic*):



pourra passer outre ce niveau de sécurité. La dernière possibilité est alors de contrôler les rubans et barres d'accès rapides comme nous le verrons beaucoup beaucoup plus loin dans le présent document. Mais dès lors il faudra faire attention à ne pas se bloquer soi-même (en

laissant au moins dans le nouveau ruban créé sur mesure le boutond d'accès à VBA qui de toute façon est normalement protégé par un mot de passe connu par l'administrateur seul!).

11.11 Barre d'outils personnalisée

11.11.1 MS Access 2003 et antérieur

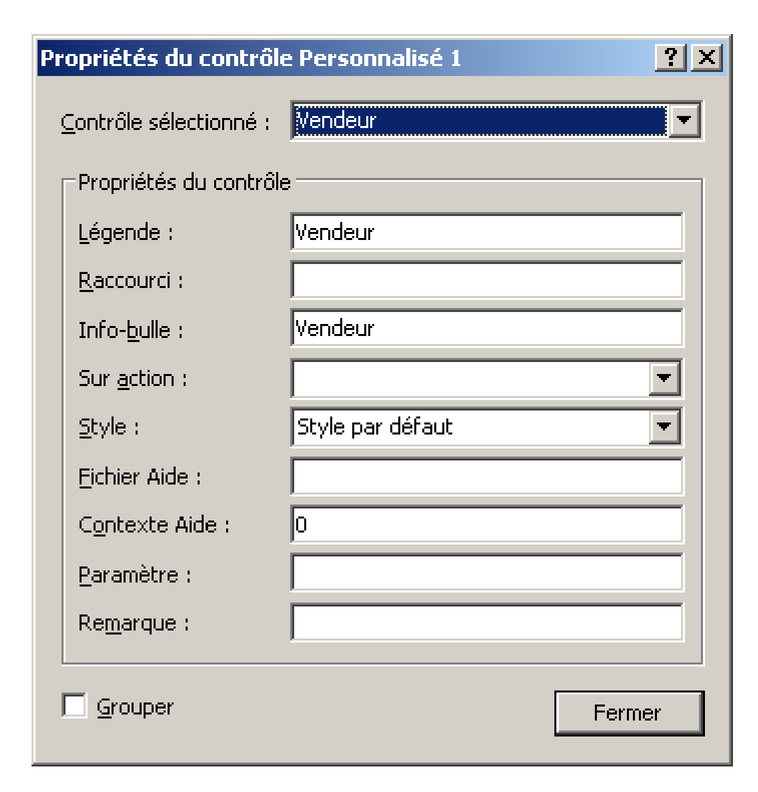
Vous avez également la possibilité d'affecter une barre de menus spécifique destinée aux utilisateurs de votre application.

Pour ce faire, il faut comme pour tout logiciel de la suite MS Office créer une barre d'outils personnalisée avec les composants que l'on souhaite (boutons ou/et menus) tel que par exemple (…):



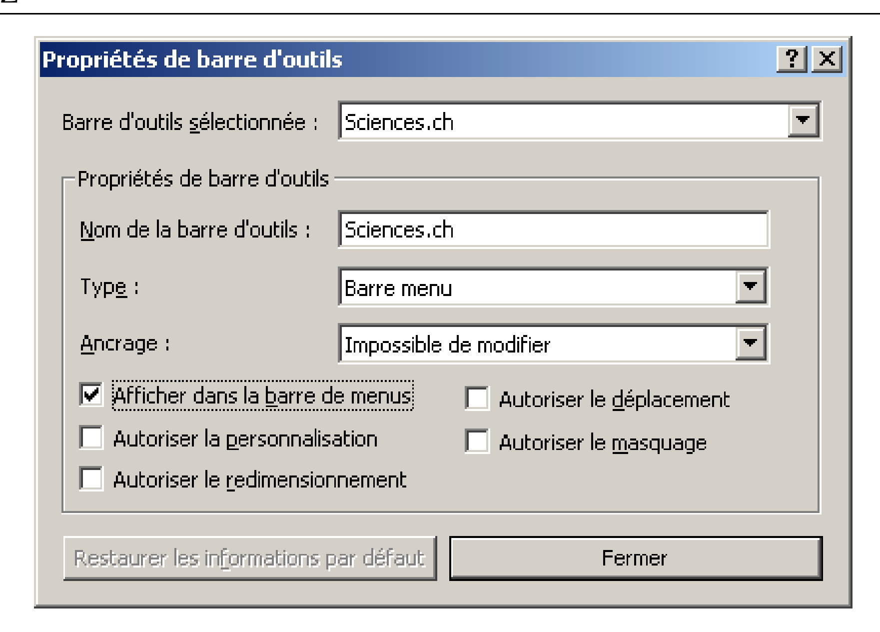
Remarque: la barre d'outils est créée dans le fichier MDB courant (ce qui veut dire que cette barre apparaît seulement à l'ouverture de la base courante) !

Si vous souhaitez exécuter une macro spécifique lors du clic d'un des boutons insérés dans la nouvelle barre d'outils, vous devez effectuer un clic droit sur le bouton en question et sélectionner *Propriétés*. Vous aurez à l'écran:



L'option importante ici est *Sur Action*. Il s'agit de sélectionner la macro qui devra être exécutée lors du clic sur votre bouton.

Dans la fenêtre *Personnaliser/Barre d'outils* cliquez sur le bouton *Propriétés* et ensuite choisissez les options comme ci-dessous (nom de la barre d'outils mis à part…):



Validez ces modifications et retournez dans *Outils/Démarrage* et dans la liste déroulante "Barre de menus" sélectionnez votre nouvelle barre de menus.

Fermez et rouvrez votre base de données. Observez le résultat…

Vous pouvez également pour chaque formulaire de votre base de données spécifier une barre d'outils particulière qui doit s'afficher lorsque le formulaire intéressé est ouvert (voir dans les propriétés du formulaire).

11.11.2 MS Access 2007 et ultérieur

Dans MS Access 2007 et ultérieur tout ce qui est relatif aux barres d'outils s'est considérablement compliqué. La raison du respect des standards XML expliquant cet état de fait, Microsoft aurait pu faire l'effort de mettre un outil simple de personnalisation à disposition des développeurs de base de donnés.

Le sujet est devenu suffisamment vaste pour qu'un chapitre à peu près exhaustif prenne entre 60 et 90 pages A4. Nous renvoyons dès lors le lecteur pour l'instant sur les PDFs téléchargeable disponibles sur www.developpez.com et nous allons montrer dans la présent document uniquement les points très très fréquemment demandés par les employés en entreprise.

Je me refuse par contre de traiter des rubans associés aux formulaires car selon c'est une ânerie car ne fonctionnant pas sur toutes les plates-formes ne comportant pas la version cliente complète de MS Office Access.

11.11.2.1 Masquer le ruban au démarrage de MS Access 2007

Commençons par la méthode pour masquer complètement les rubans au démarrage de MS Access 2007.

Pour cela il faut créer une table nommée *USysRibbons* avec une colonne nommée *RibbonName* de type *Texte* et une autre colonne nommée *RibbonXML* de type *Memo*.

Il suffit ensuite de mettre dans cette table les valeurs suivantes:

Nom colonne	Contenu
RibbonName	HideTheRibbon
RibbonXML	<CustomUI xmlns="http://schemas.microsoft.com/office/2006/01/CustomUI"> <ribbon startFromScratch="true"/> </CustomUI>

et ensuite il suffit dans les options de la base de données active d'associer le ruban *HideTheRibbon* comme ruban de démarrage.

Évidemment, il est possible de nommer le ruban autrement que *HideTheRibbon* et d'y mettre un autre code XML.

Par exemple le code XML suivant:

```
<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui">
  <ribbon startFromScratch="true">
    <tabs>
      <tab id="tabTest" label="Test"visible="true">
        <group id="grpTest" label="Test">
          <button id="idTest" label="Test sans icone standard" size="large"/>
          <button idMso="Paste" label="Test avec icone standard" size="large"/>
        </group>
      </tab>
    </tabs>
  </ribbon>
</customUI>
```

11.11.2.2 Masquer le ruban au démarrage de MS Access 2010

Dans MS Access 2010 la méthode pour complétement masquer tout ruban consiste à exécuter la commande VBA suivant lors du chargement du formulaire d'accueil:

```
DoCmd.ShowToolbar "Ribbon", acToolbarNo
```

12 Finitions élémentaires

Le but ici est de présenter les finitions élémentaires au niveau sécurité de votre base de données ainsi que les interactions élémentaires entre cette dernière est quelques logiciels de la suite MS Office.

12.1 Compactage

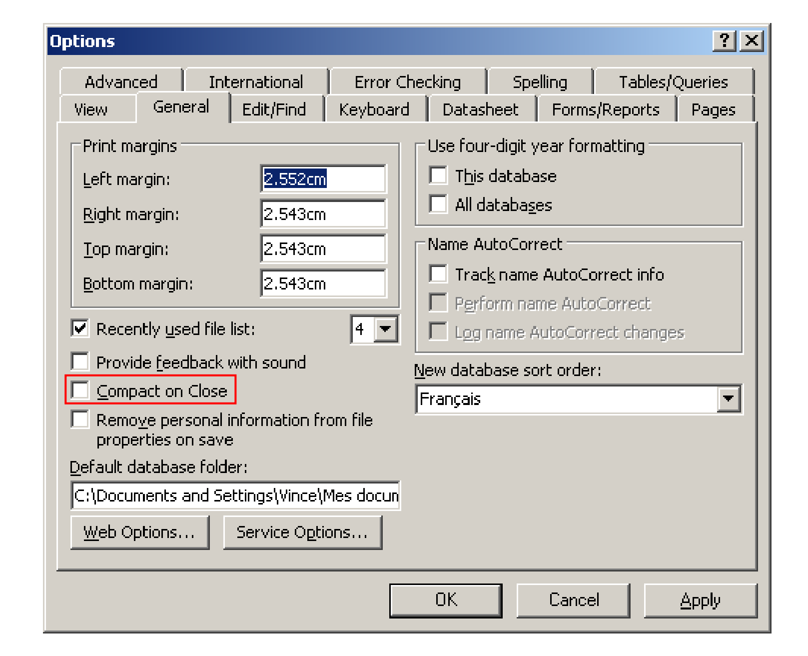
Souvent, les fichiers MS Access ont des tailles non négligeables. Sachez que l'on peut compacter (compresser) la base de données en allant dans:

Outils/Utilitaire de base de données/Compacter une base donnée

Pour info le record de ce que j'ai vu de pire jusqu'à aujourd'hui est le passage d'une base de 76 Mo à 1.3 Mo…

Au fait, il faudrait comprimer la base de données systématiquement après chaque modification… Mais ne vous inquiétez pas, Microsoft a pensé à tout ! Allez dans:

Outils/Options/Général/Compacter à la fermeture



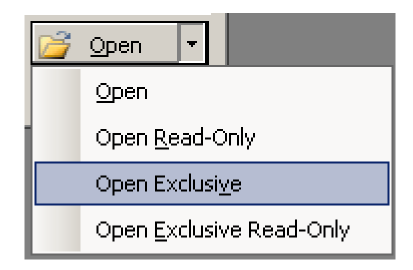
12.2 Protection par mot de passe

Pour définir (ou supprimer) un mot de passe d'accès à la base vous devez aller dans:

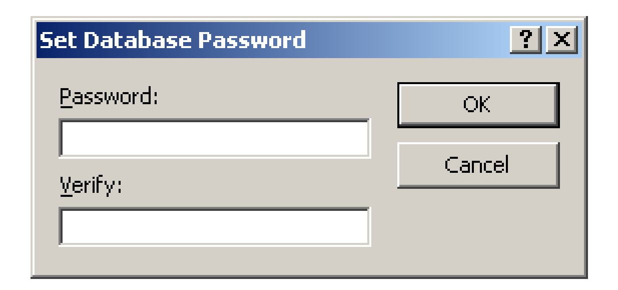
Outils/Sécurité/Définir le mot de passe de la base de donnée

mais pour faire cela vous devez d'abord avoir ouvert la base de données en mode *Exclusif*:

Fichier/Ouvrir en mode exclusif (option cachée sur le bouton Ouvrir)



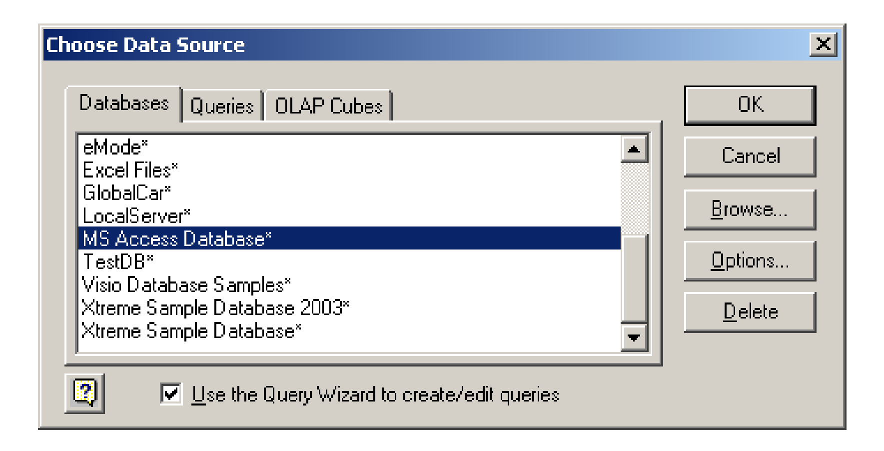
Vous pourrez ensuite définir le mot de passer comme indiqué précédemment. Apparaîtra la boîte e dialogue suivante:



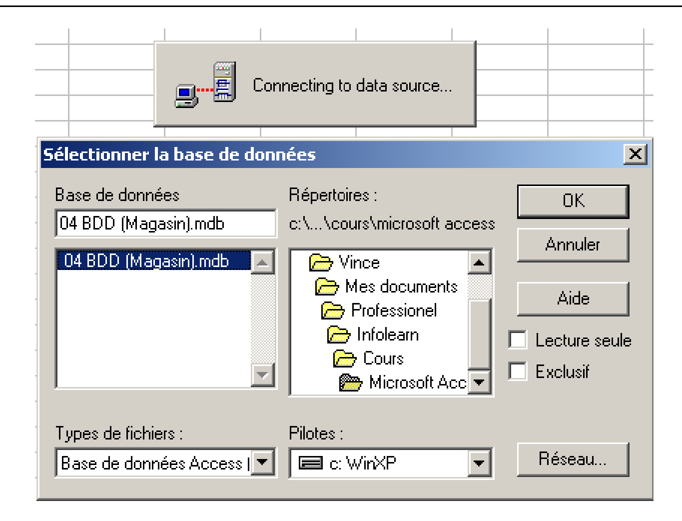
et quand vous fermez et rouvrez votre base de données, MS Access vous demandera à chaque fois de saisir le mot de passe.

12.3 MS Query

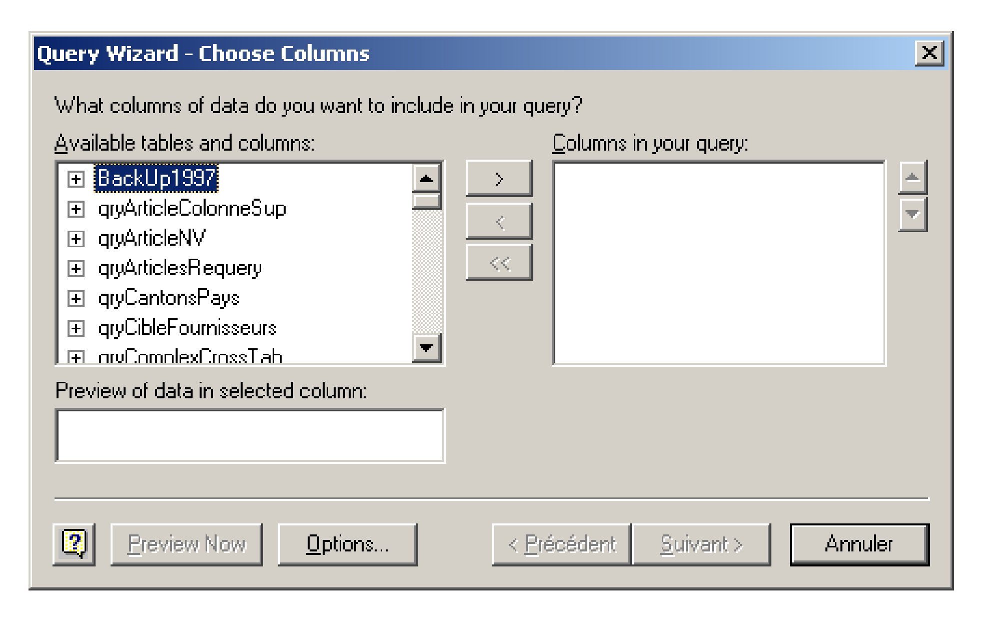
Le but de cet exercice consiste à importer des données de la dernière version de notre base dans MS Excel. Pour ceci, nous utiliserons dans MS Excel MS Query. Vous pouvez alors dans ce logiciel aller dans *Données/Données externes/Créer une requête* puis apparaît:



quand vous cliquez sur *OK* vous pouvez aller chercher la base intéressée:



Veuillez après avoir cliqué sur *OK* choisir les champs qui vous intéressent:



Le reste fait l'objet d'un cours MS Excel !

12.4 Publipostage

Le but ici, est de vous introduire à la technique de publipostage dans MS Word depuis une base MS Access. Personnellement je considère cela cependant plus comme un cours MS Word que comme un cours MS Access mais bon cela peut toujours être utile…

La raison s'explique par le fait, que bien que la mise en place des "champs de fusion" soit simple, toutes les options qu'il existe dans MS Word autour de ces dernières sont assez conséquentes et pourraient faire l'objet d'un cours de 3 heures pour des non-initiés à ce logiciel.

Vous voilà avertis. Cependant si vous tenez à voir du MS Word et laisser de côté MS Access pendant un moment, votre formateur se tient à votre disposition pour vous montrer le fonctionnement du publipostage (et ses fonctions avancées).

Remarque: Malheureusement, MS Access doit être fermé pour faire du publipostage depuis MS Word.

13 Macros

Une macro est composée d'actions, chaque action correspond à une tâche: lorsque vous exécutez la macro, MS Access exécute automatiquement les actions qu'elle contient. Certaines de ces actions, plus complexes, permettent d'afficher des boîtes de dialogue, de tester la réponse fournie par l'utilisateur, d'afficher une barre de menus personnalisée… et de développer ainsi une application autonome sans que vous ayez besoin de programmer (bien que les actions fassent référence à des instructions du langage VBA).

Disons tout de même que les macros ont l'énorme mérite d'ouvrir la porte aux non programmeurs, en leur permettant de commencer sans trop de difficultés. Elles sont donc réservées aux nouveaux venus tout en étant interdites à tout développeur professionnel confirmé ! Profitez-en donc !

Enumérons quelques limites des macros:

1. Pas de débogueur: test de variables, arrêt sur changement de valeur…
2. Pas de contrôle d'erreur: s'il y un problème la macro plante sans espoir
3. Pas d'instruction de boucles (Do… While, For… Next, etc.)
4. Expression conditionnelle permettant de sauter une groupe d'instructions sans rapport avec un If… Then… Else… ou mieux encore, un Select Case…
5. Pas de programmation structurée: procédures Sub(), Function()
6. Pas de paramètres dans l'appel d'une macro
7. Pas de variables globales non plus, pour passer une valeur.
8. Encore moins de programmation Objet
9. Aucun accès aux références extérieures: pas d'automation OLE pour piloter Word, Excel, Outlook et les autres
10. De nombreuses limitations: string SQL de 256 caractères (au lieu de 32'000 en VBA)
11. etc.

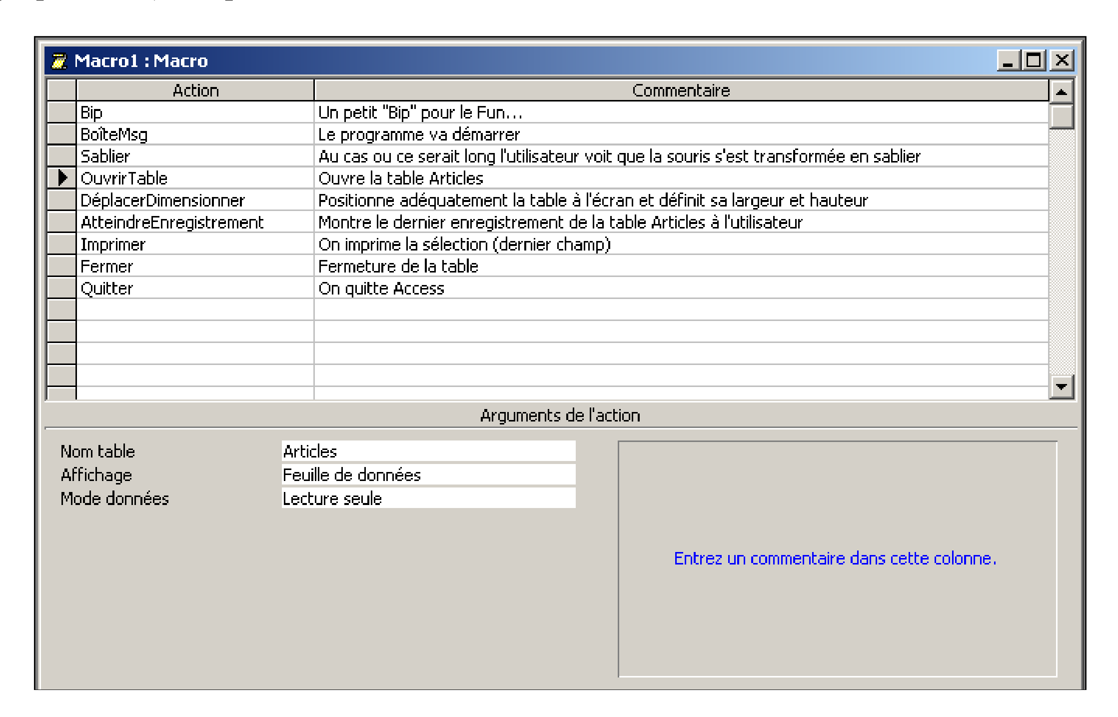
Il faut savoir enfin que MS Access comporte une command *Outils/Macro/Convertir les macros en Visual Basic*. A Consommer sans modération. C'est un des meilleurs moyens d'apprendre le VBA.

Remarque: les macros sont la très fréquemment affectées à des boutons posés dans des formulaires et pris de la barre d'outils "contrôles".

13.1 Macros simples

Pour créer une macro il suffit d'aller dans l'explorateur d'objets de cliquer sur la catégorie *Macros* et de cliquer ensuite le bouton *Nouveau*. Le type d'actions les plus usitées sont, dans l'ordre, les suivantes:

Sablier, Bip, Boitemsg, OuvrirFormulaire, OuvrirEtat, Imprimer, OuvrirRequete, Fermer (état, formulaire ou requête et autres), AppliquerFiltre, AfficherTousEnreg (contraire de AppliquerFiltre), DéplacerDimmensionner, ExécuterCode, ArrêtMacro, Quitter…

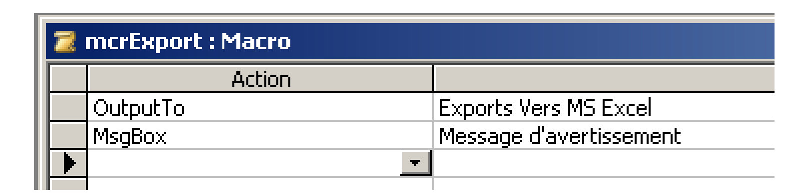


Chaque action comporte des arguments qui seront passés en paramètres à la macro VBA. Un descriptif des effets de l'action est donné par Microsoft. Vous pouvez ajouter et supprimer des lignes dans la Macro comme vous le feriez pour une table normale (bouton droit de la souris).

Avant de s'attaquer à des macros complexes il faut avoir fait !

1. Les exercices des requêtes (voir plus haut) avec les macros d'export et import de MS Excel

2. Les macros de messages (MsgBox)

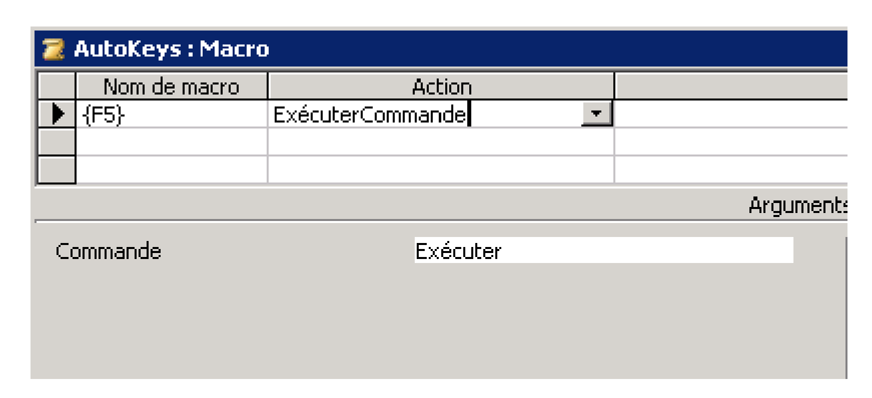


13.1.1 Exécution de requêtes en mode création

Lors de la création ou modification de requêtes en mode création, de nombreux utilisateurs m'aiment pas utiliser Ctrl+Point ou Ctrl+Virgule pour passer d'un affichage à l'autre (deux boutons c'est pas génial...).

Il existe une méthode cependant pour créer un raccourci (pouvant être utilisé à d'autres fins!) avec une seul touche pour exécuter une requête en mode création.

Cette méthode consiste dans MS Access 2003 et antérieur à créer la macro suivante et de bien la nommer *AutoKeys* (il faudra fermer et rouvrir la base pour qu'elle soit prise en compte!):



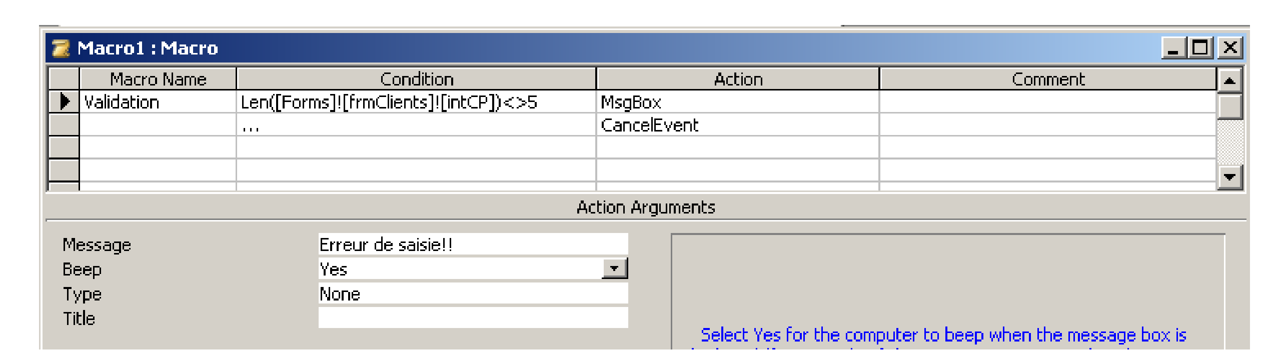
Dans MS Access 2007 et ultérieur voilà à quoi elle ressemble:



13.1.2 Contrôle de saisie simple

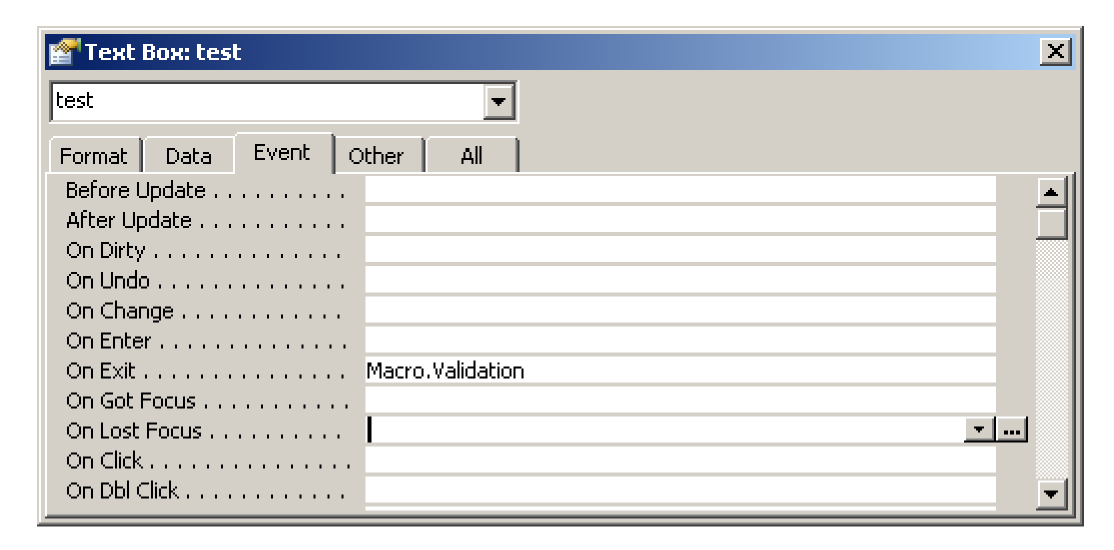
Dans cet exercice nous allons déjà introduire les macros de groupe. L'idée est de vérifier la saisie de notre collègue dans le formulaire *frmClient* au niveau du code postal *intCP* afin de vérifier que celui-ci aura bien 4 caractères.

Pour cela nous allons créer une macro nommée *Macro* de ce type:



Pour avoir fait apparaître les colonnes *Macro Name* et *Condition* nous avons cliqué sur les boutons de la barre d'outils (c'est pas utile de faire une macro groupée si jamais…)

Ensuite dans les propriétés du champ *intCP* du formulaire *frmClients* nous définissons:



et le tour est joué!

13.1.3 Import de données

Comme exercice de rappel, et pour voir si vous avez tout compris jusque-là, vous allez faire un petit exercice (durée d'une heure environ).

Créez un fichier *.xls à l'aide de MS Excel dans lequel vous saisirez quelques données (et seulement quelques colonnes aussi) qui correspondent à une des tables de notre base magasin. Ensuite, l'objectif est:

1. D'importer la table XL dans MS Access
2. D'éliminer les éventuelles lignes vides dans la table importée à l'aide d'une requête de suppression
3. De vérifier et éliminer les doublons dans la table importée à l'aide de l'assistant de requête de recherche des doublons
4. De vérifier si dans la table importée il n'y a pas des informations qui se trouveraient déjà à l'identique dans la table cible (avec l'assistant requête de non-correspondance).
5. En ayant éliminé les données redondantes du point (4), importez les nouvelles données dans la table cible à l'aide d'une requête d'ajout
6. Videz la table importée précédemment (à l'aide d'une requête de suppression)
7. Exportez la table vidée en écrasant l'ancien fichier XL existant (avec une action de la macro bien sûr !)
8. Ouvrez la table cible utilisée plus haute avec l'aide de la macro et imprimez la automatiquement (toujours avec la macro)

A la fin, rien ne doit transparaitre à l'écran pour notre utilisateur, seulement quelques messages d'avertissement et une feuille imprimée.

Bonne chance !

13.1.4 Contrôles sur formulaires

Considérons le formulaire suivant (l'esthétique n'est pas le sujet ici!):



Nous souhaitons à l'aide d'une macro à l'ouverture de ce formulaire, désactiver (ou cacher) le bouton de calcul du CA tant que les champs de début de période ou de fin de période ne contiennent pas une information.

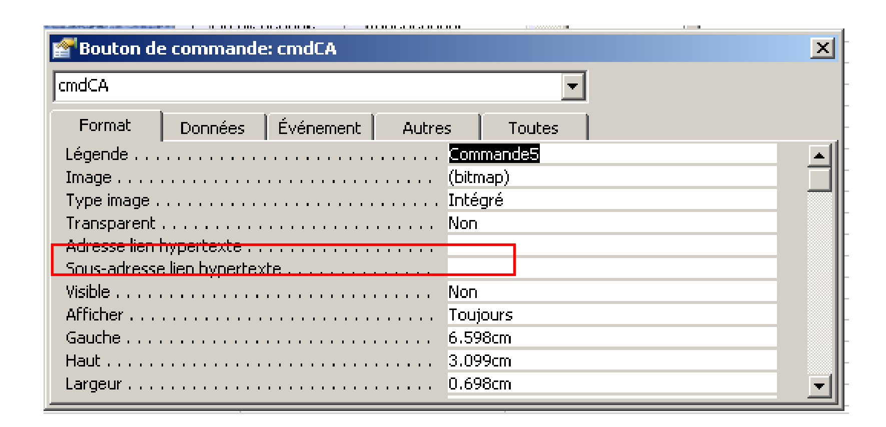
Les données disponibles sont les suivantes:

1. Le formulaire se nomme *frmCAPeriode*

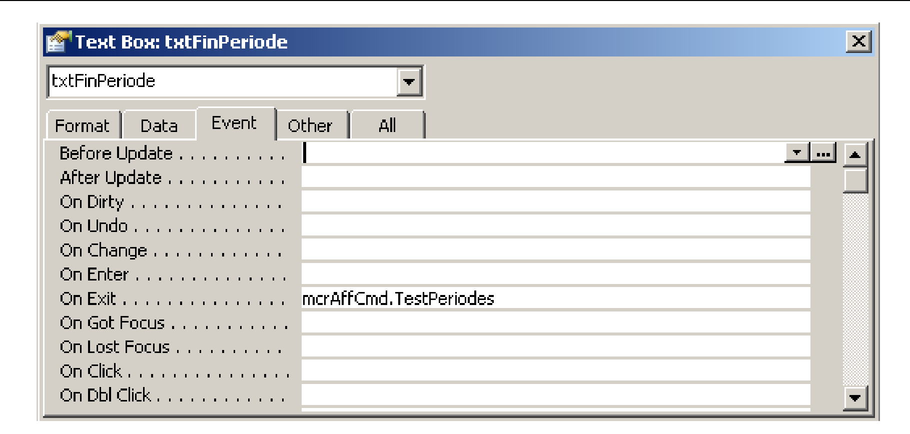
2. Les deux champs de dates se nomment respectivement *txtDebutPeriode* et *txtFinPeriode*

3. Le bouton se nomme *cmdCA*

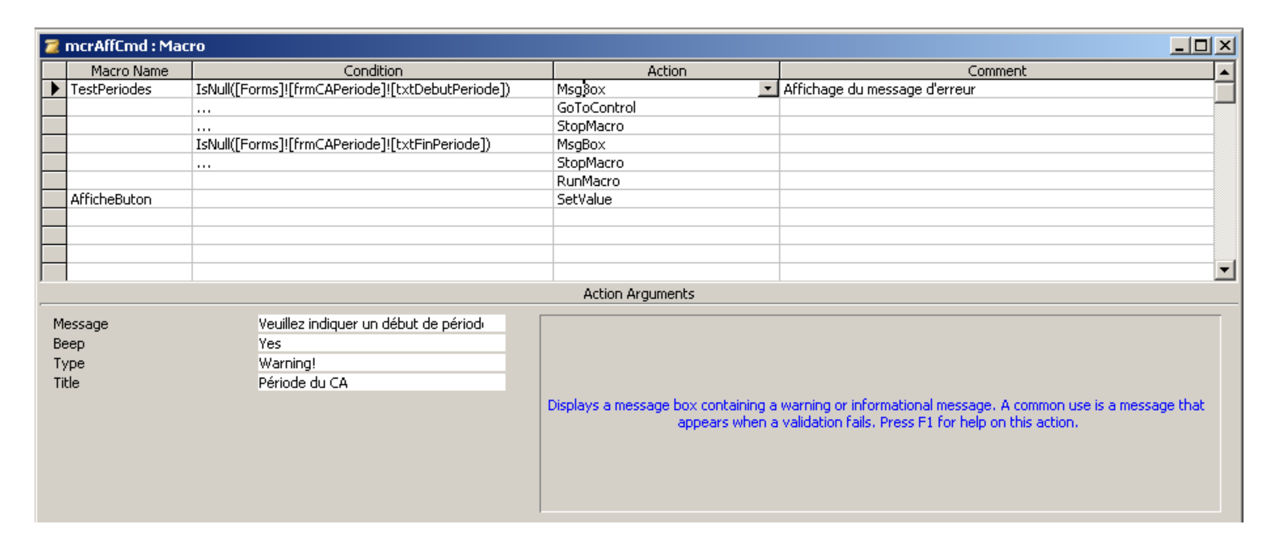
Nous allons faire en sorte que le bouton *cmdCA* soit caché par défaut à l'ouverture du formulaire:



Nous voulons lors de la perte du focus du champ *txtFinPeriode*, exécuter une macro nommée *mcrAffCmd* qui nous activera l'affichage du bouton si et seulement si les deux champs de date ont un contenu non nul!



La macro satisfaisant l'exemple est de la forme suivante:



Nous voyons que dans un premier temps (sous-macro *TestPeriodes*), celle-ci effectue un contrôle du contenu du champ de début de période. Si celui-ci est vide, l'action *GoToControl* donne le focus au champ au *Control Name*: *txtDebutPeriode*. Après quoi la macro s'arrête.

Idem pour le cham de fin de période.

Si les deux champs de dates ont un contenu, alors séquentiellement, MS Access exécutera la sous-macro *AfficheBouton* dont l'action *SetValue* a pour *Item*:

[Forms]![frmCAPeriode]![cmdCA].[Visible]

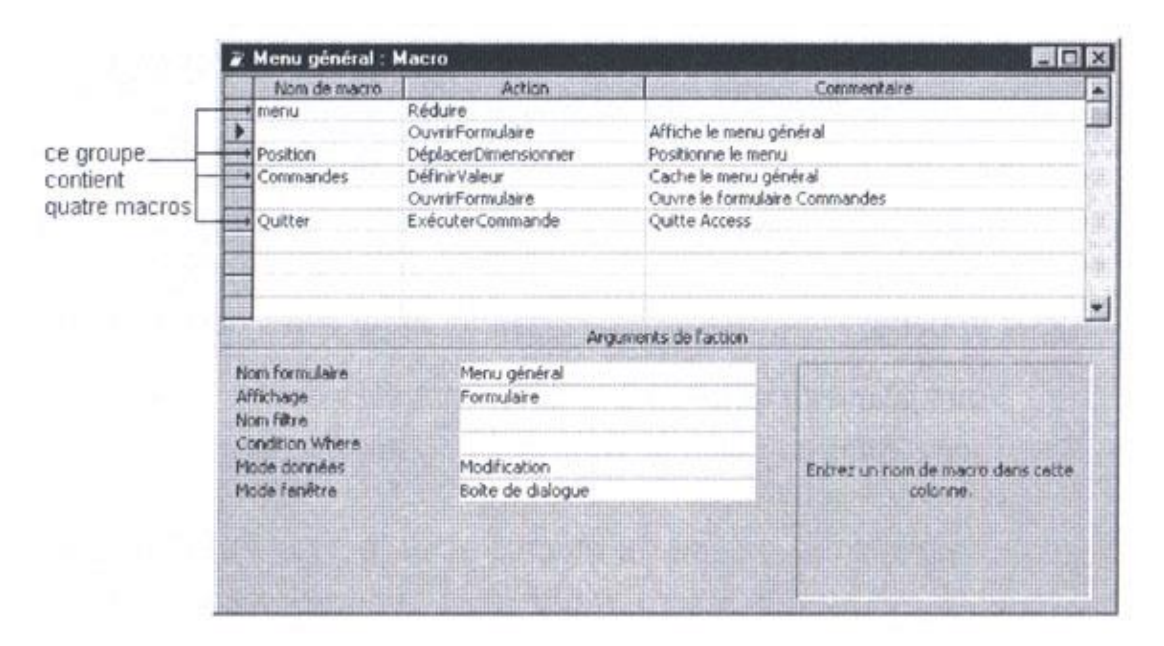
et pour valeur: *True*

Ainsi le bouton d'exécution de notre requête, ne sera visible que si la date de début ET la date de fin ne sont pas vides de contenu.

Attention!!! Pensez toujours à lâcher le focus sur un élément que vous allez masquer en utilisant l'action GoToControl (sur un autre contrôle à choix).

13.2 Groupe de macros

Vous avez la possibilité de créer une suite de groupe de macros comme on l'a représenté dans la capture d'écran ci-dessous. Pour faire apparaître cette fenêtre, cliquez sur le bouton après avoir créé une nouvelle macro.



Vous pouvez bien évidemment enregistrer le groupe de macros comme on enregistre un état, formulaire, table en cliquant simplement sur la petite disquette habituelle.

Pour faire appel à une des macros du groupe à partir d'un objet quelconque, vous devrez alors la dénommer sous la forme: *NomduGroupe.NomDeLaMacro*.

Pour cette raison, donnez à vos groupes et macros des noms suffisamment explicites mais pas trop longs.

Pour exécuter un macro appartenant autonome, utilisez la commande:

Outils-Macro-Exécuter une Macro

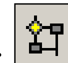
ou sinon à partir d'une fenêtre macro en mode création cliquez sur le petit point d'exclamation:

Access exécute alors chaque action de la fenêtre Macro, les unes à la suite des autres. Il s'arrête lorsqu'il rencontre une ligne vierge, l'instruction *ArretMacro* (ou *ArretMacro*) ou si la macro affiche une boite de message.

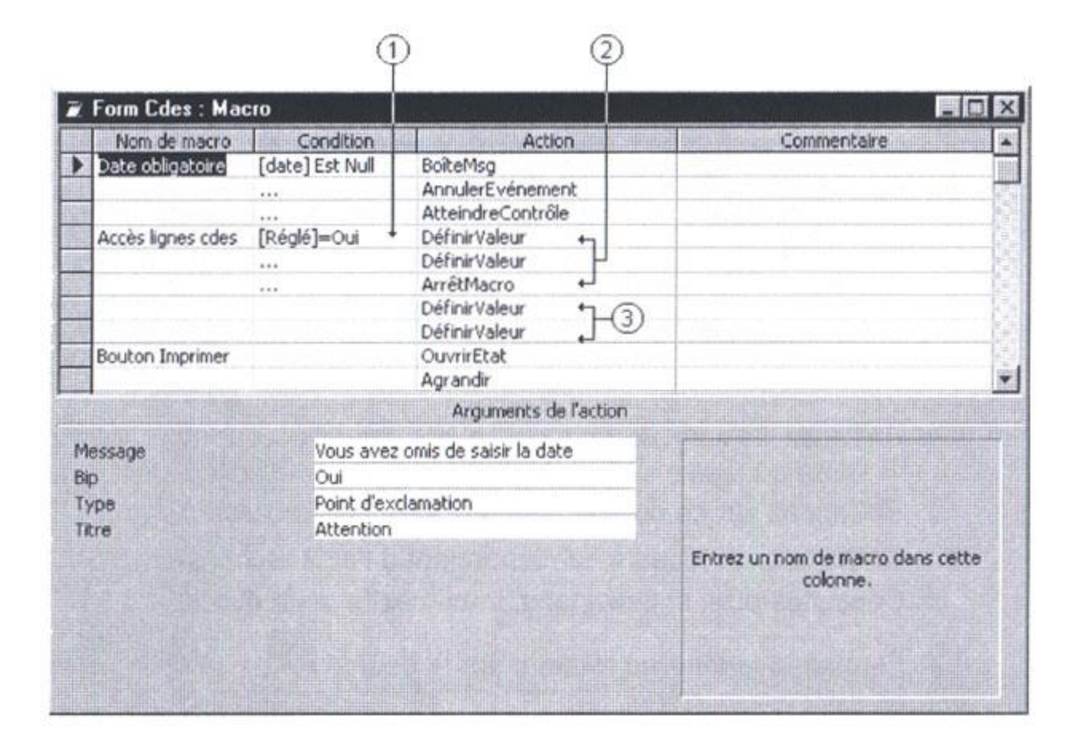
Pour exécuter une macro à l'ouverture d'une base, appelez-la AutoExec (pour éviter d'exécuter cette macro à l'ouverture de la base, maintenez la touche *Shift* enfoncée pendant l'ouverture de la base).

Afin d'analyser le déroulement d'une macro, vous pouvez l'exécuter pas à pas. Pour ce faire, affichez la macro en mode création et cliquez sur l'outil puis exécutez la macro.

13.3 Groupe conditionnel de macros

Actions en fonction de conditions: 

Si vous cliquez sur ce bouton vous faites apparaître à la fois le mode Groupe des macros et une nouvelle colonne de conditions comme représenté sur la figure de la page suivante:



(1) Précisez l'expression logique permettant à Access de tester la condition.

(2) Précisez la ou les actions à réaliser si la condition est vraie. S'il y a plusieurs actions, tapez des points de suspension sur chaque ligne de la colonne "Condition"; sur la ligne suivante, insérez l'action *ArrêtMacro* pour interrompre la macro une fois les premières actions effectuées.

(3) Précisez ensuite les actions à réaliser si la condition est fausse.

Remarques:

Si la condition n'est pas vérifiée, Access exécute l'action située sur la première ligne qui ne contient pas de points de suspension!

Pour faire référence à un champ ou à un contrôle ne se trouvant pas sur l'objet actif, faites précéder le nom du champ du type et du nom de l'objet en les séparant par un point d'exclamation (appelé "opérateur d'identification).

Exemples:

Formulaires![Clients]![Code]

se réfère au champ "Code" du formulaire "Clients".

Etats![Adresses]![Nom]

se réfère au champ "Nom" de l'état "Adresses"

Pour faire référence à la propriété d'un champ ou d'un objet, faites suivre son identification d'un point et du nom de la propriété. Exemples:

[Code].Visible

se réfère à la propriété "Visible" du champ "Code" du formulaire actif

Formulaires![Clients].Visible

se réfère à la propriété "Visible" du formulaire "Clients"

Formulaires![Clients]![Code].Visible

se réfère à la propriété du champ "Code" du formulaire "Clients"

Si la base de données comporte du code Visual Basic Application, en l'enregistrant comme fichier MDE on permet de compiler tous les modules, de supprimer tout le code source modifiable et de compacter la base de données de destination.

Le code Visual Basic sera toujours exécuté, mais il ne pourra plus être visualisé ni modifié ; de plus, la base de données prendra moins de place car le code source est supprimé et l'utilisation de la mémoire est optimisée, ce qui augmente les performances.

La conversion en MDE nécessite cependant un zéro faute dans votre code sinon la conversion ne réussira pas !

L'enregistrement de la base de données comme fichier MDE, empêche les opérations suivantes:

- Afficher, modifier ou créer des formulaires, des états ou des modules en mode Création.
- Ajouter, supprimer ou modifier des références aux bases de données.
- Modifier le code Visual Basic , car le fichier MDE ne contient aucun code source.
- Changer le nom du projet VBA de la base de données à l'aide de la boîte de dialogue Options.
- Importer ou exporter des formulaires, des états ou des modules. Par contre, il est toujours possible d'importer ou d'exporter des tables, des requêtes ou des macros à partir ou vers des bases de données non MDE.

Attention:

Il faut absolument effectuer une **copie** de la base de données d'origine. Si l'on désire modifier ultérieurement la structure d'un formulaire, d'un état ou d'un module d'une base de données

enregistrée au format MDE, on doit effectuer ces modifications **dans la base de données d'origine**, puis l'enregistrer à nouveau comme fichier MDE.

C'est pour cette raison qu'il faut éviter d'enregistrer comme fichier MDE une base de données comportant des **tables**… On ne pourra pas simplement modifier la structure d'un formulaire, d'un état ou d'un module dans la version d'origine, car les données des tables qu'elle contient ne sont plus à jour ! Il est plutôt conseillé de choisir la base de données frontale qui contient toute la structure de gestion (requêtes, formules, états, modules) pour l'enregistrer comme fichier MDE, alors que la base qui contient les tables se contentera d'être liée.

Marche à suivre:

1. Fermer la base de données. Dans un environnement multi-utilisateur, tous les utilisateurs doivent fermer la base de données.

2. Menu "Outils" – "Utilitaires de base de données" – "Créer fichier MDE".

3. Dans la boîte de dialogue "Base de données à enregistrer comme MDE", indiquer la base de données que l'on désire enregistrer comme fichier MDE et cliquer sur "Créer MDE".

4. Dans la boîte de dialogue "Enregistrer MDE comme", indiquer le **nom**, l'**unité** et le **dossier** de la base de données.

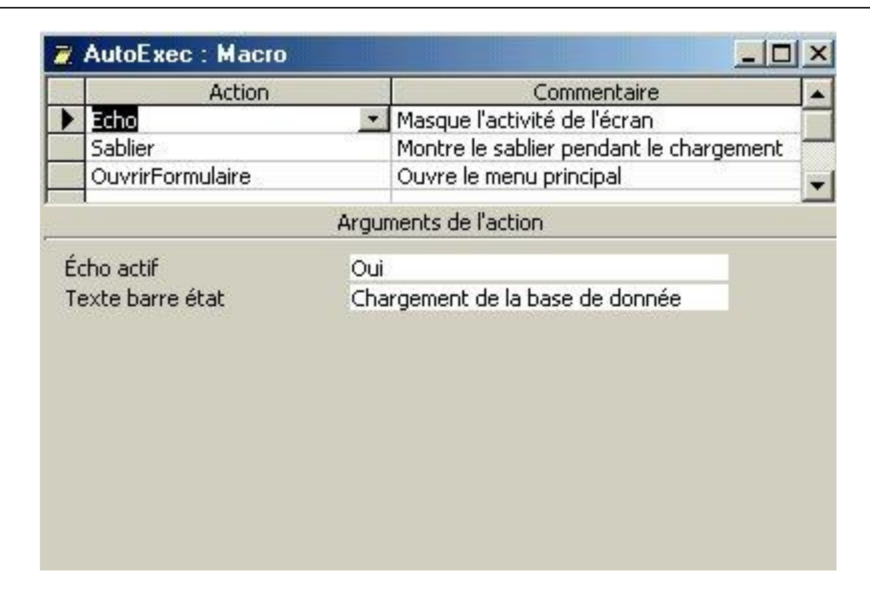
13.4 Macro AutoExec

Vous pouvez utiliser une macro spéciale appelée "AutoExec" pour exécuter une action ou une série d'actions lorsque votre base de données est ouverte pour la première fois. Quand vous ouvrez une base de données, Microsoft Access recherche une macro de ce nom et, s'il en trouve une, l'exécute automatiquement.

Procédure à suivre:

Crée une macro qui ouvrira votre formulaire principal, enregistrer la macro sous le nom "AutoExec". Aucun autre nom, comme "Mon AutoExec" ou "autoexec" par exemple n'est autorisé. Alors **attention à la casse**!!

Votre macro ressemblera à la figure suivante:



Cette macro AutoExec est de la plus haute importance car nous la retrouverons lors de l'étude des options avancées du travail en mode multi-utilisateurs.

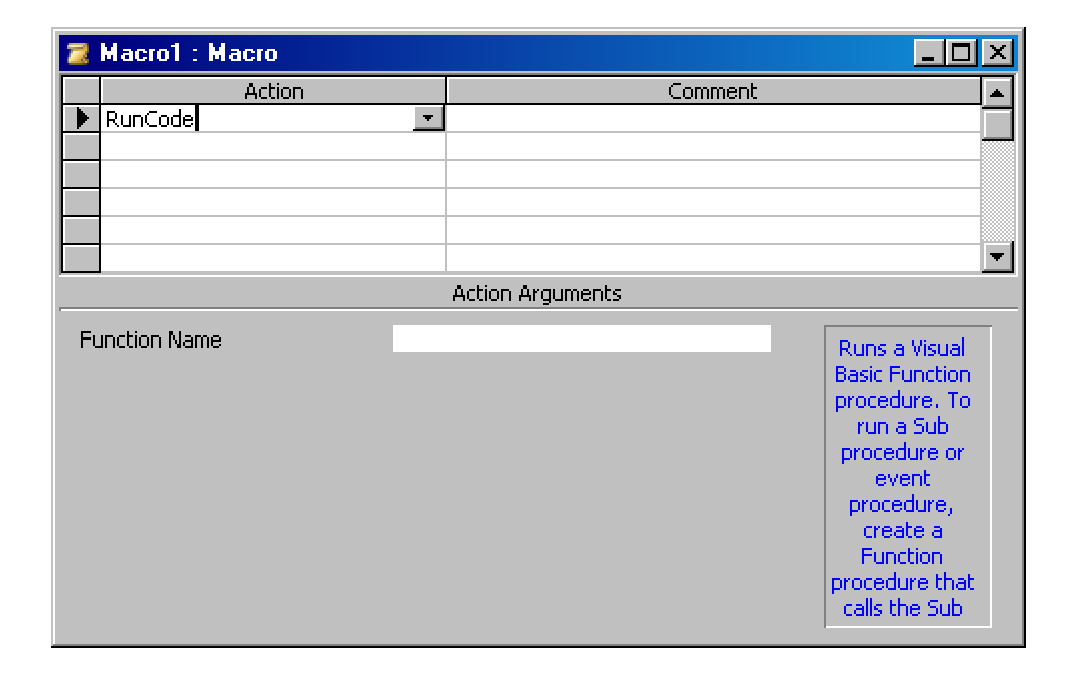
13.5 Macros Run Code

Une action important pour beaucoup d'employés reprenant des bases de données d'anciens collègues qui maîtrisaient le code VBA ou utilisant des base de données développées par des entreprises externes souhaitent parfois utiliser des codes VBA via des macros simples.

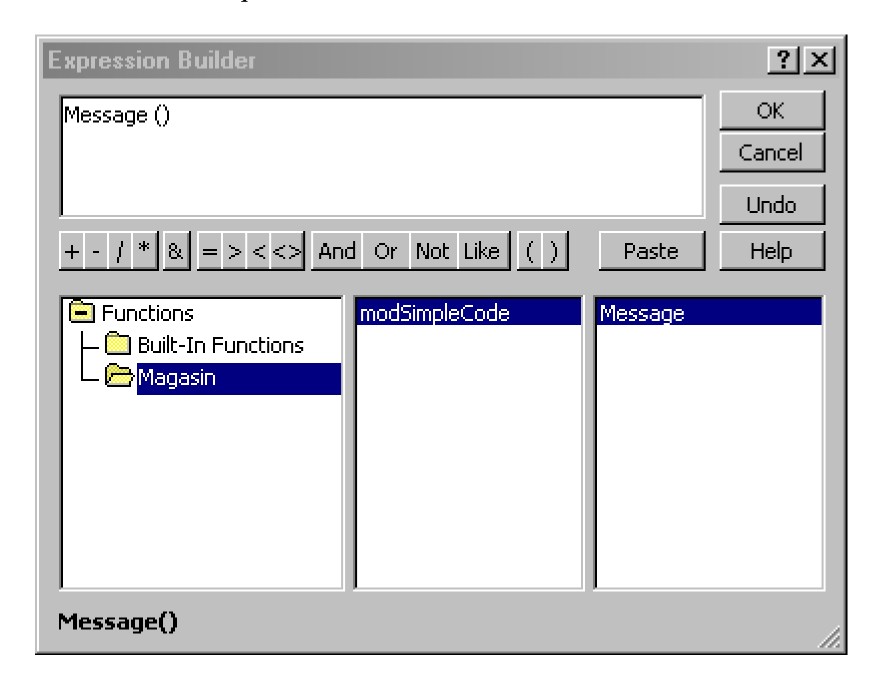
La commande réservée à cette action est *RunCode*. Elle a une limitation non négligeable dont il faut se souvenir: elle ne peut pas exécuter des routines (sub) mais uniquement des fonctions (function) VBA.

Ainsi, si vous avez écrit une routine (sub), il faudra soit la transformer en fonction, soit créer une fonction qui fait appel à la routine.

Nous avons alors:



Et dans function name nous cliquons sur le bouton d'assistance:

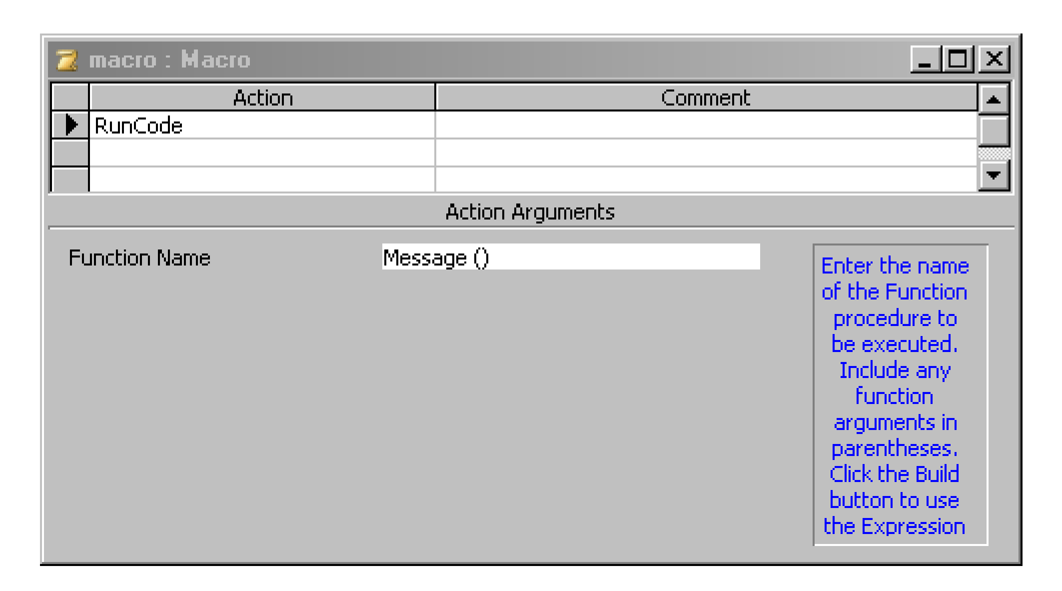


où *Message* est une simple fonction VBA Access utilisée pour l'exemple:

```
Option Compare Database

Function Message()
    MsgBox "Hello Word"
End Function
```

Nous avons alors:



Ce qui donnera à l'exécution de la macro:



13.6 Macros Import/Export

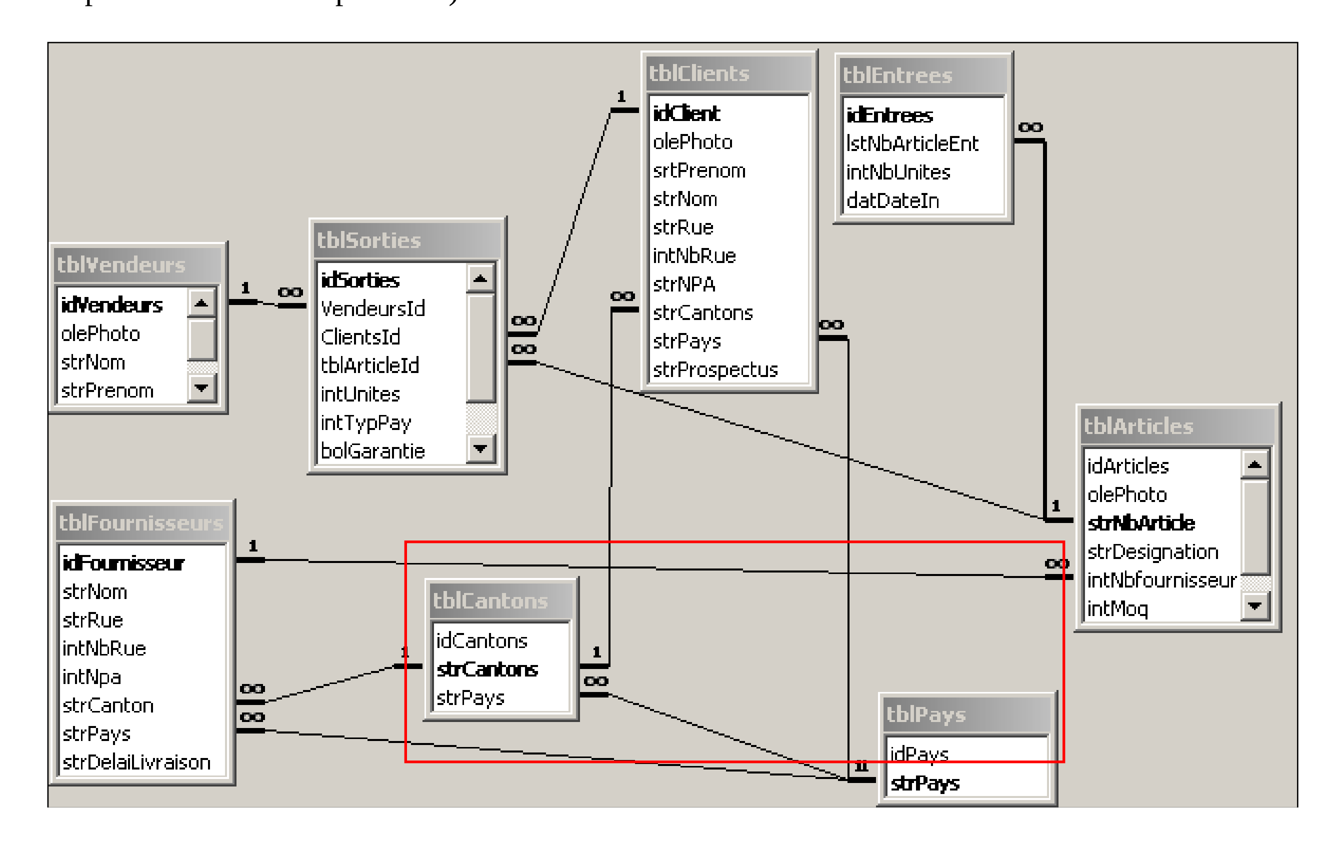
Nous avons déjà fait antérieurement dans ce document lors de notre étude des requêtes des macros qui avaient pour but d'exporter ou d'importer des données de ou vers MS Excel. Afin de nous assurer que ces concepts soient maîtrisés, nous vous demandons de créer une macro *mcr Export* capable d'exporter la table *tblSorties* dans un fichier MS Excel qui devra être enregistré sur la racine de votre disque C:\ et s'ouvrir automatiquement.

Nous vous demandons également de créer une macro *mcrImport* qui aura pour rôle d'importer des données du fichier MS Excel *NouveauxVendeurs.xls* dans la table des vendeurs (attention de vérifier que tous les champs soient compatibles… n'oubliez pas que nous aimons bien insérer des pièges dans ce support de cours !)

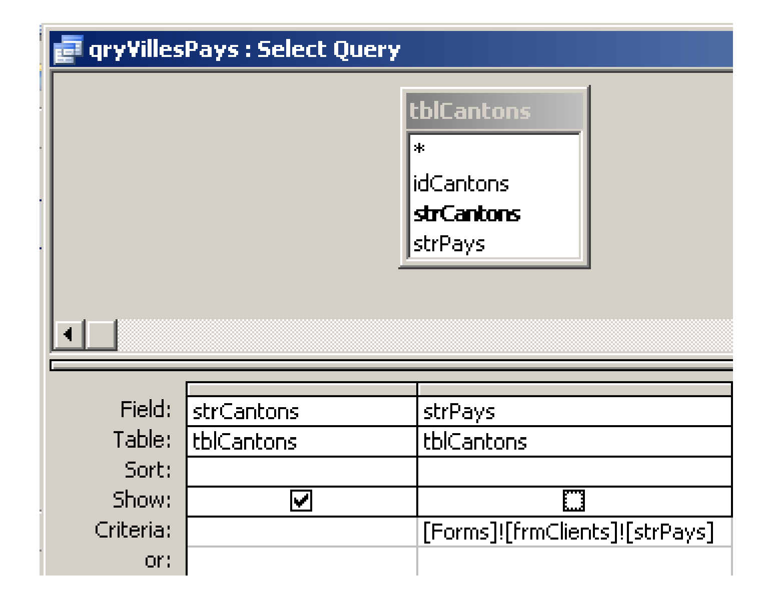
13.7 Macro ReQuery

Nous allons maintenant créer dans notre formulaire *frmClients* une macro qui remplit automatiquement le champ des cantons en fonction de ce qui aura été choisi dans le champ des pays avec la macro *ReQuery* qui est très importante dans le monde MS Access!

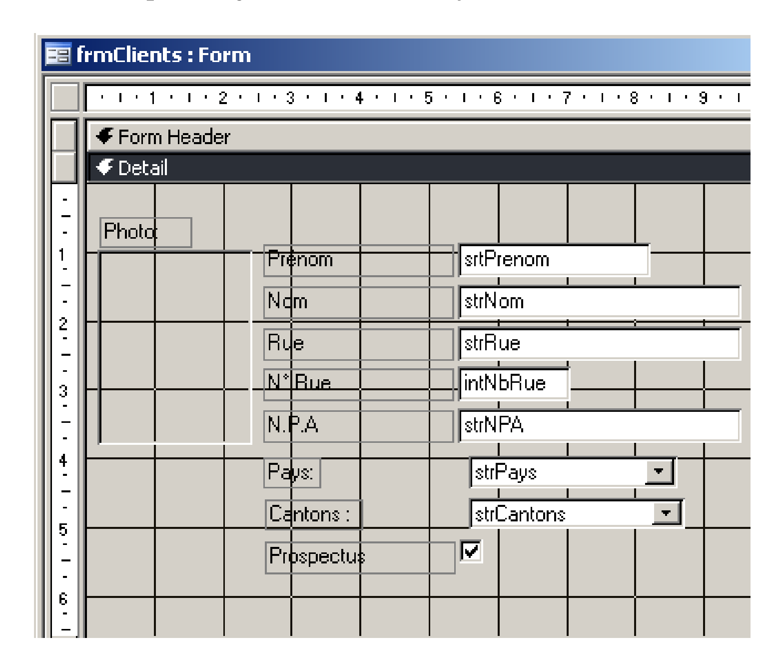
Vous devez d'abord vous arranger à retravailler votre base de manière à avoir au niveau des tables (particulièrement les tables *tblPays* et *tblCantons*) le schéma suivant (veillez à remplir un peu les deux tables précitées):



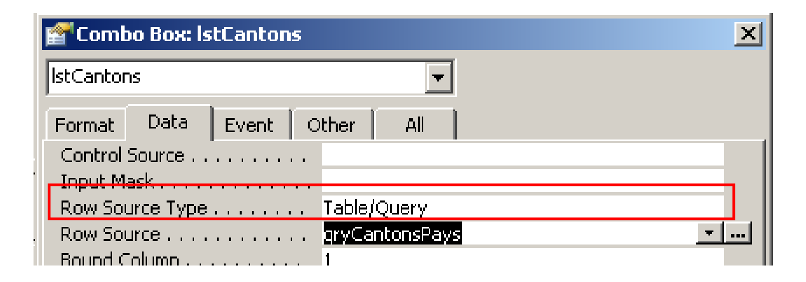
Et ensuite créez d'abord une requête (*qryCantonsPays*) telle que présentée ci-dessous:



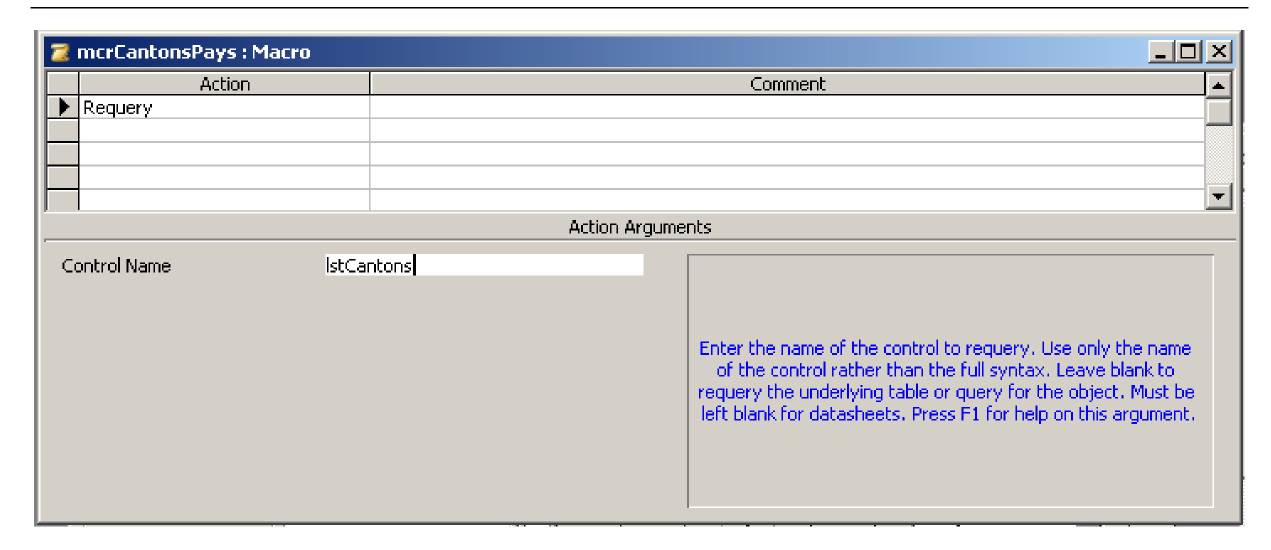
avec comme critère le champ *strPays* du formulaire *frmClients*:



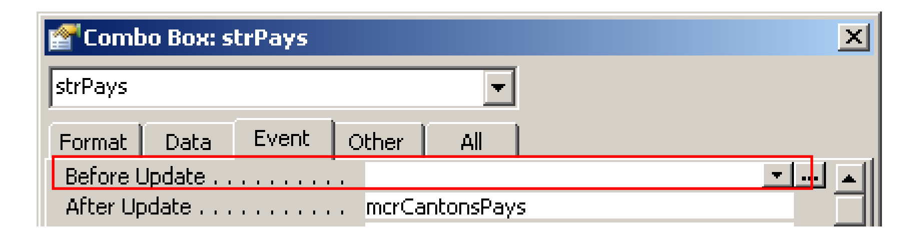
Dans le formulaire *frmClients* (ci-dessus), veillez à avoir la propriété suivante pour le champ *strCantons* (il va chercher les données dans la requête que nous venons de créer):



Créez aussi une macro nommée *mcrCantonsPays* avec l'événement *Requery* (en français: *Actualiser*):



et ensuite veillez à avoir pour la combo box *strPays* la propriété suivant:



et voilà qui est fait.

La macro requery est également très utile lorsque l'on souhaite que les listes déroulantes affichent leur contenu uniquement au fur et à mesure de la saisie clavier de l'utilisateur. Le problème cependant c'est qu'il faut alors faire appel uniquement à du VBA et du SQL en dure. Raison pour laquelle nous traiterons de ce sujet dans le chapitre sur le VBA

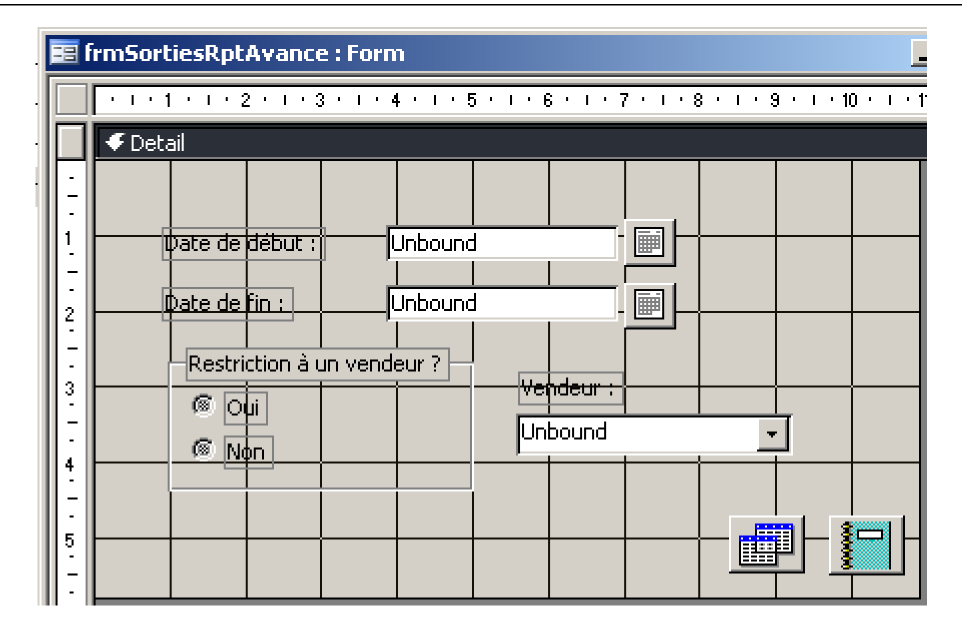
13.8 Groupe de macros simple

Cet exercice est la suite de celui que nous avions fait sur le formulaire interagissant avec une requête et un état (voir exercice 44). Copiez le formulaire et nommez-le *frmSortiesRptAvance*.

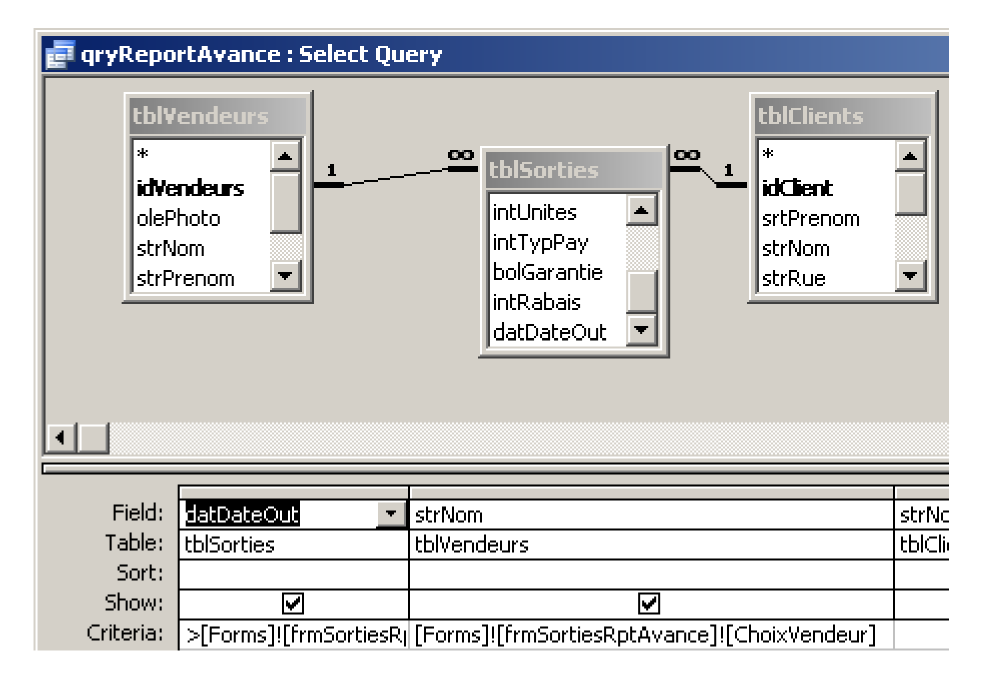
1. Ajoutez-y une liste déroulante (nommée *lstNomVendeurs*) contenant le nom des vendeurs (liez-y une requête qui va chercher la liste des vendeurs. ceci vous évitera des problèmes possibles avec la clé primaire numérotée)

2. Un groupe d'options (nommé *grpChoixSynthese*) à deux valeurs *Oui*/*Non*

Voici le résultat à obtenir:



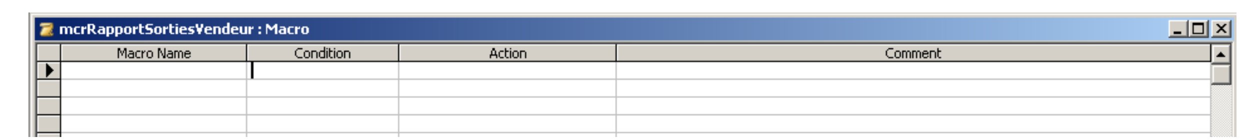
Modifiez la requête qui est rattachée à ce formulaire comme ci-dessous (attention, veuillez remplacer dans la capture d'écran ci-dessous la variable *ChoixVendeur* par *lstNomVendeurs*) :



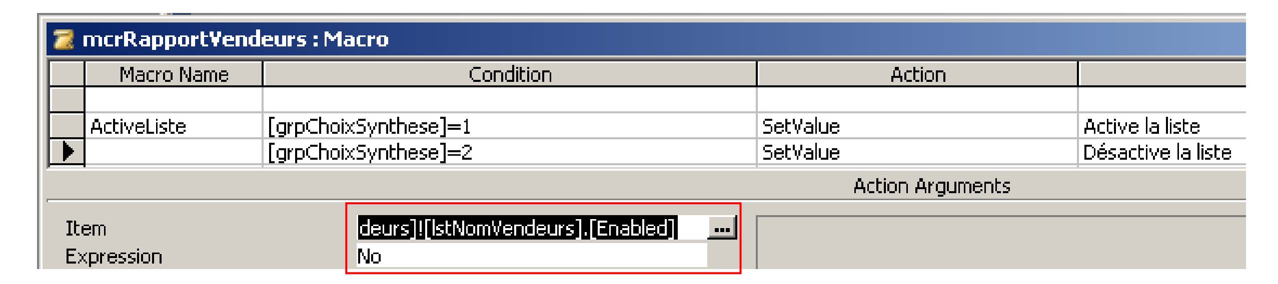
L'exemple ci-dessous marche parfaitement (au niveau de la liste des vendeurs pour l'instant) si vous avez fait tout juste jusque-là:



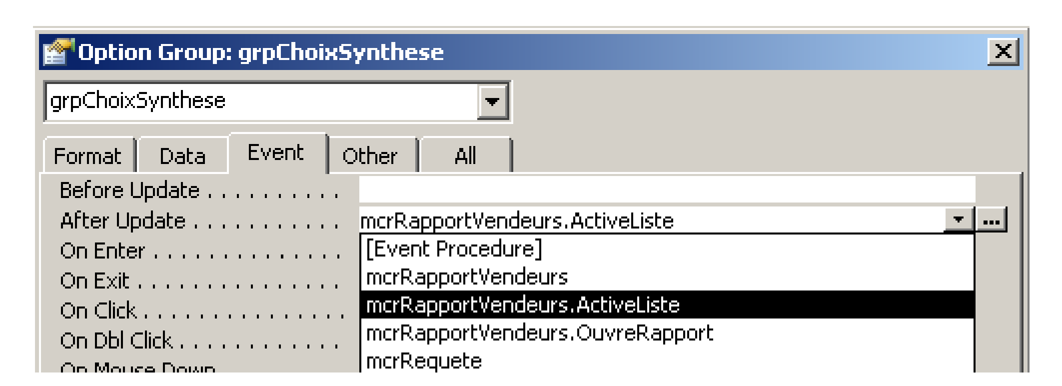
Il faut maintenant nous occuper des boutons d'option qui ont pour rôle, d'activer ou de désactiver le choix des vendeurs. Créez une nouvelle macro conditionnelle vide nommée *mcrRapportVendeurs*:



Cette première étape permet d'activer/désactiver la liste déroulante du choix des vendeurs (Expression vaut *Yes* pour le premier et *No* pour le second):



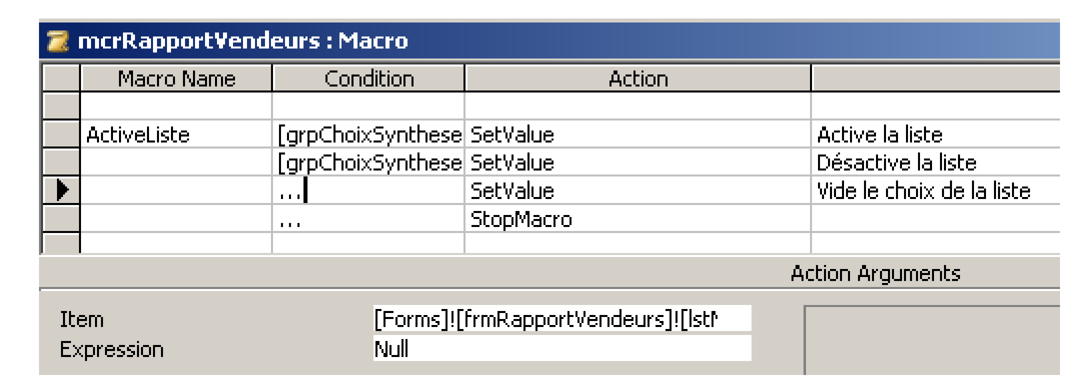
et rattachez à l'événement *AfterUpdate* du groupe d'options:



Une fois ceci fait, votre formulaire devrait fonctionner. Ainsi lorsque l'utilisateur clique sur *Oui* la liste des vendeurs s'active et inversement.

Mais si nous ouvrons la requête, celle-ci va quand même prendre le nom du vendeur dans la liste même si celle-ci est désactivée. Il nous faut donc vider cette liste en rajoutant quelques actions comme ci-dessous (attention les noms des champs diffèrent !!!).

Remarque: prenez garde à ne pas copier ce qui est écrit sur les captures d'écran ci-dessous sans réfléchir à ce que vous faites !

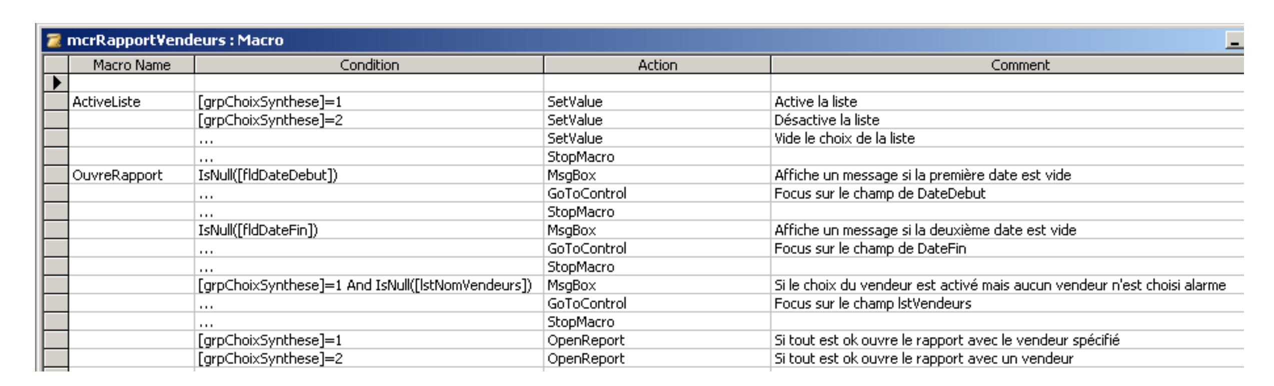


Il nous faut donc maintenant modifier en conséquence le critère de la requête relativement aux champs du nom de vendeur. Ainsi, comme nous l'avons déjà vu, si le paramètre est vide, il convient d'écrire (c'est une possibilité parmi celles que nous avons vu jusqu'ici dans le cours et pas nécessairement la plus simple !):

[Forms]![frmSortiesRptAvance]![lstNomVendeurs] OR
[Forms]![frmSortiesRptAvance]![lstNomVendeurs] is Null

Maintenant il nous faut dans ce groupe, une dernière macro qui va s'exécuter lorsque l'utilisateur voudra visualiser le rapport. Cette macro devra vérifier si les champs des dates ne sont pas vides et le cas échéant si un nom de vendeur aurait été choisi.

Voici à quoi doit ressembler la macro (vous la faites avec le formateur):



mcrRapportVendeurs : Macro

Macro Name	Condition	Action	Comment
ActiveListe	[grpChoixSynthese]=1	SetValue	Active la liste
	[grpChoixSynthese]=2	SetValue	Désactive la liste
	...	SetValue	Vide le choix de la liste
	...	StopMacro	
OuvreRapport	IsNull([fldDateDebut])	MsgBox	Affiche un message si la première date est vide
	...	GoToControl	Focus sur le champ de DateDebut
	...	StopMacro	
	IsNull([fldDateFin])	MsgBox	Affiche un message si la deuxième date est vide
	...	GoToControl	Focus sur le champ de DateFin
	...	StopMacro	
	[grpChoixSynthese]=1 And IsNull([lstNomVendeurs])	MsgBox	Si le choix du vendeur est activé mais aucun vendeur n'est choisi alarme
	...	GoToControl	Focus sur le champ lstVendeurs
	...	StopMacro	
	[grpChoixSynthese]=1	OpenReport	Si tout est ok ouvre le rapport avec le vendeur spécifié
	[grpChoixSynthese]=2	OpenReport	Si tout est ok ouvre le rapport avec un vendeur

Avec l'assistant création de bouton, créez après un bouton lié à la macro:

mcrRapportVendeurs.OuvreRapport

13.9 Groupe de macros complexe

Refaites de même mais avec la table *tblEntrées* en faisant en sorte que l'utilisateur puisse:

1. Choisir un intervalle de dates pour les entrées

2. Choisir les articles et/ou les fournisseurs

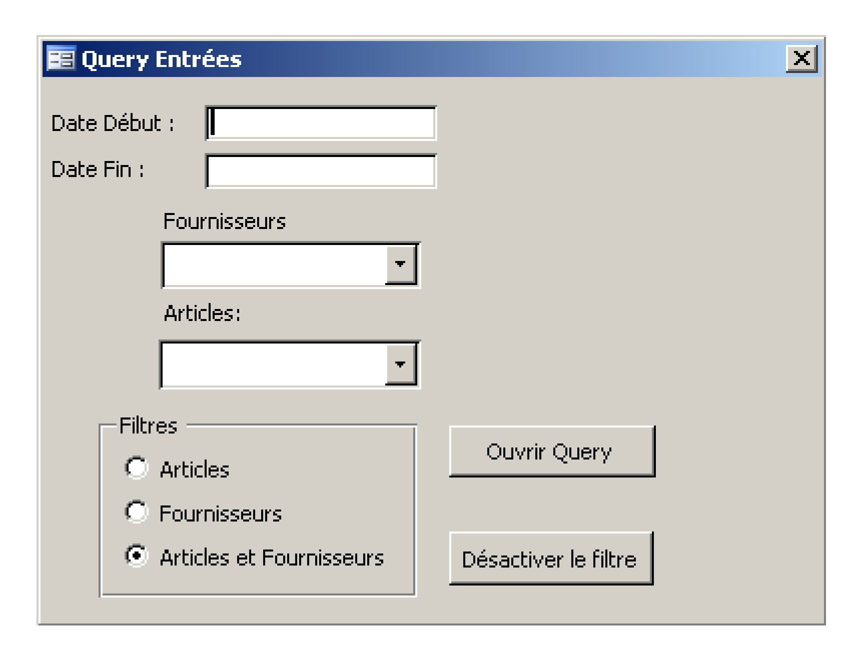
3. Lorsqu'un fournisseur est activé que la liste des articles n'affiche que ceux du fournisseur sélectionnée (requery)

4. Activer ou désactiver le filtre

Remarque: il faudra créer plusieurs requêtes ainsi que plusieurs macros ainsi qu'un formulaire. Nous vous laissons le choix des noms.

Le résultat peut ressembler à quelque chose comme cela (modulo vos goûts esthétiques personnels) outre les requêtes qui sont considérées comme triviales:

1. Le formulaire:

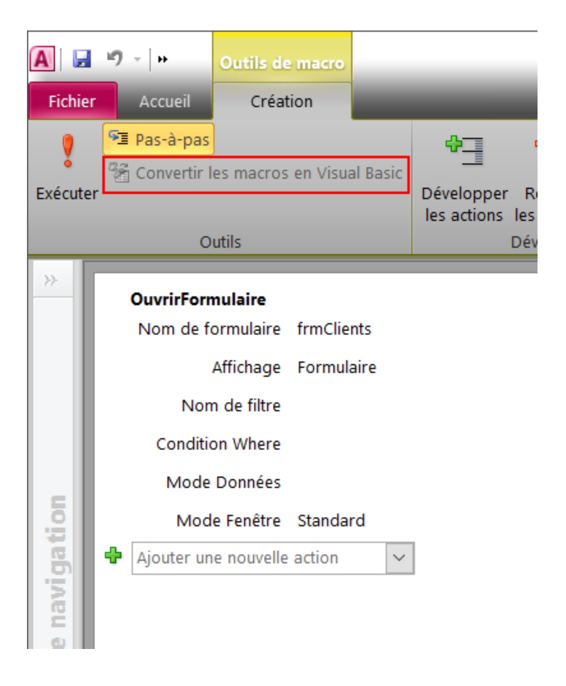


2. La macro principale:

Macro Name	Condition	Action	Comment
ActiveListe	[grpGroupeSynthese]=1	SetValue	active la liste articles
	...	SetValue	désactive la liste fournisseurs
	...	SetValue	vide la liste fournisseurs
	...	StopMacro	
	[grpGroupeSynthese]=2	SetValue	désactive la liste articles
	...	SetValue	active la liste des fournisseurs
	...	SetValue	vide la liste des articles
	...	StopMacro	
	[grpGroupeSynthese]=3	SetValue	
	...	SetValue	
	...	StopMacro	
FilterOff		SetValue	
		SetValue	
		SetValue	
		SetValue	
		StopMacro	
OuvrirQuery	IsNull([fldDateD])	MsgBox	
	...	StopMacro	
	IsNull([fldDateF])	MsgBox	
	...	StopMacro	
	([lstFournisseurs].[Enabled]=No And [lstArticles].[Enabled]=No)	OpenQuery	
	...	StopMacro	
	[grpGroupeSynthese]=1 And IsNull([lstArticles])	MsgBox	
	...	StopMacro	
	[grpGroupeSynthese]=2 And IsNull([lstFournisseurs])	MsgBox	
	...	StopMacro	
	[grpGroupeSynthese]=3 And (IsNull([lstFournisseurs]) Or IsNull([lstArticles]))	MsgBox	
	...	StopMacro	
		OpenQuery	

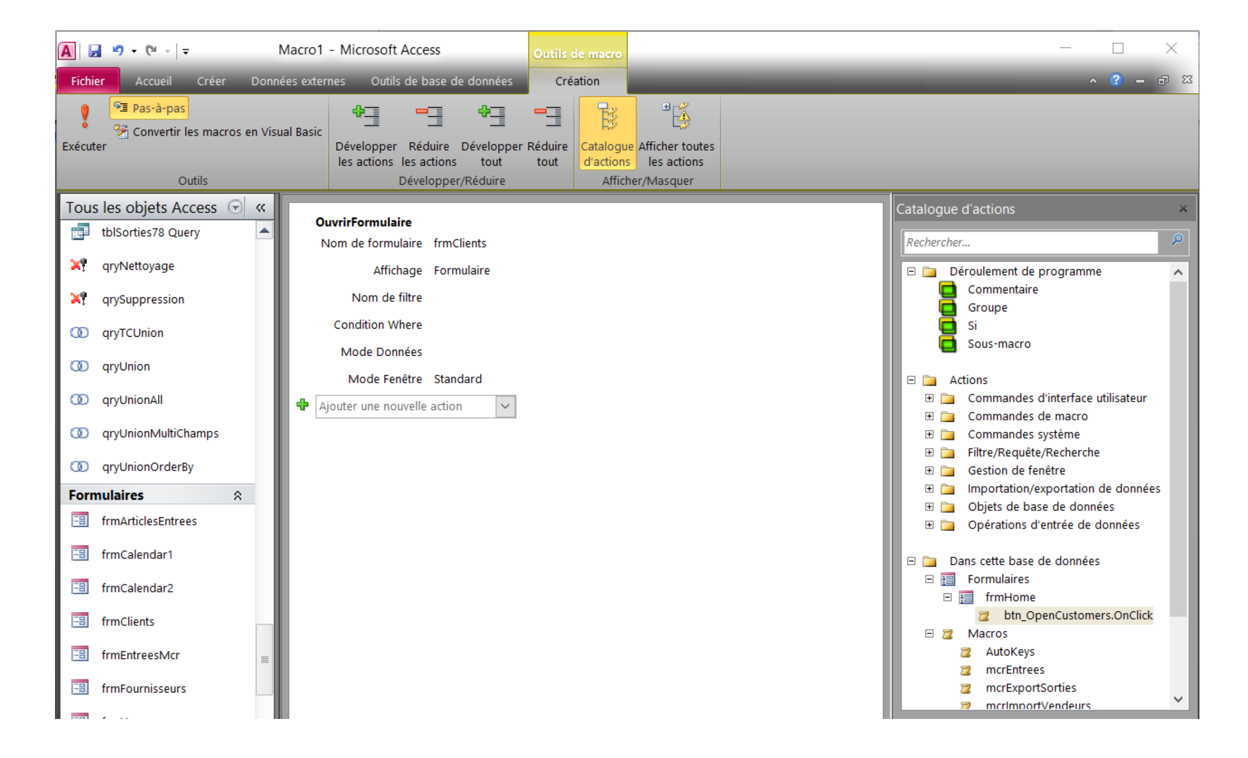
13.1 Conversion de Macros en VBA

La conversion de Macros en code VBA était facile avant 2007. Depuis… c'est devenu un peu plus tordu surtout pour les macros générées automatiquement par le logiciel. Effectivement, voici la macro d'un bouton de formulaire résultat de la création par l'assistant de bouton de Microsoft Access:



Comme nous pouvons le voir, le bouton *Convertir les macros en Visual Basic* est grisé. Alors comment? comment? comment faire????

Eh bien c'est subtil mais… simple. Il suffit de créer une nouvelle Macro et ensuite dans la volet de droite aller chercher dans la base de données en cours, le formulaire ET le bouton qui nous intéressent:



et comme nous pouvons le voir, le bouton *Convertir les macos en Visual Basic* n'est plus grisé!

14 Théorie internet / intranet

Si vous souhaitez que le personnel de votre entreprise ait accès au contenu de la base (ou des tables) sans utiliser la notion de "partage" incluse dans MS Access, vous pouvez alors publier des états-web sur l'Intranet ou Internet de votre entreprise.

Ouvrez l'objet (table ou requête uniquement) contenant les données à insérer dans la page Web puis utilisez la commande *Fichier – Exporter*.

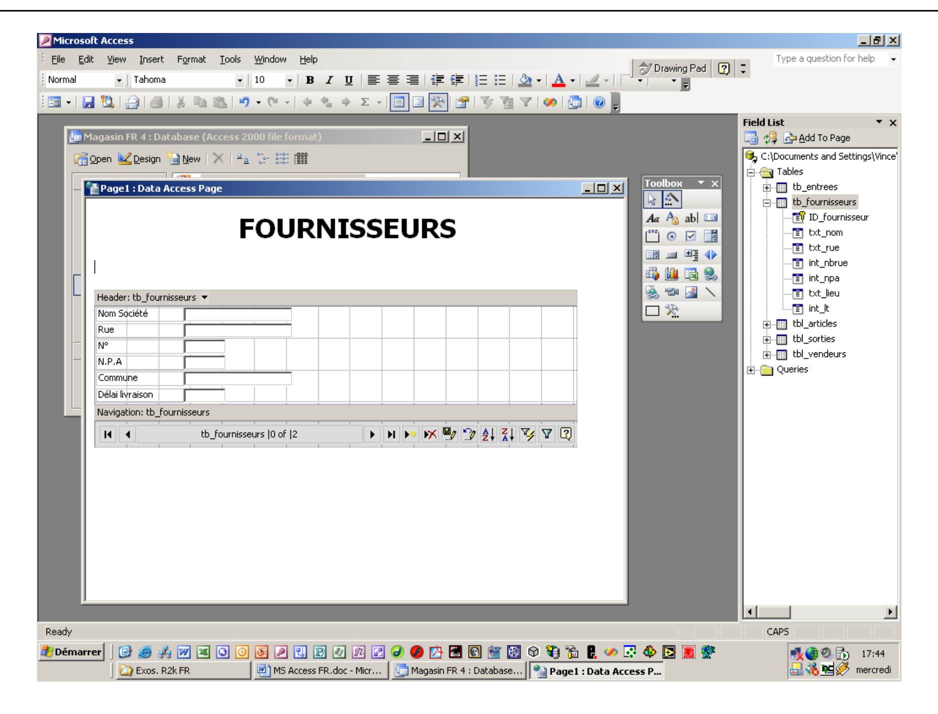
1. Modifiez si besoin est, le nom du fichier
2. Sélectionnez le type de fichier *.htm
3. Sélectionnez le dossier d'enregistrement
4. Cliquez sur enregistrer et publiez ensuite les pages de façon adéquate en utilisant Windows (dossier Web (lié FTP)).

Ce type de page ne permettra que la visualisation des données sous forme HTML simple sans interaction possible !!!

Si la page d'accès aux données doit être accessible à chaque utilisateur du réseau, placez la base de données dans un dossier partagé. Si elle doit être accessible sur un réseau Internet ou Intranet, placez-là dans un dossier sous le répertoire racine du serveur Web (\Webshare\wwwroot ou \Inetpub\wwwroot dans le IIS) qu'il faut savoir évidemment configurer et installer...

14.1 Formulaire web

Dans la fenêtre de base de données, il faut cliquer sur le bouton *Pages* dans la barre des objets puis lancez l'assistant de pages web pour créer un formulaire web simple de la table *tblFournisseurs*. Vous devez à la fin obtenir le résultat suivant:

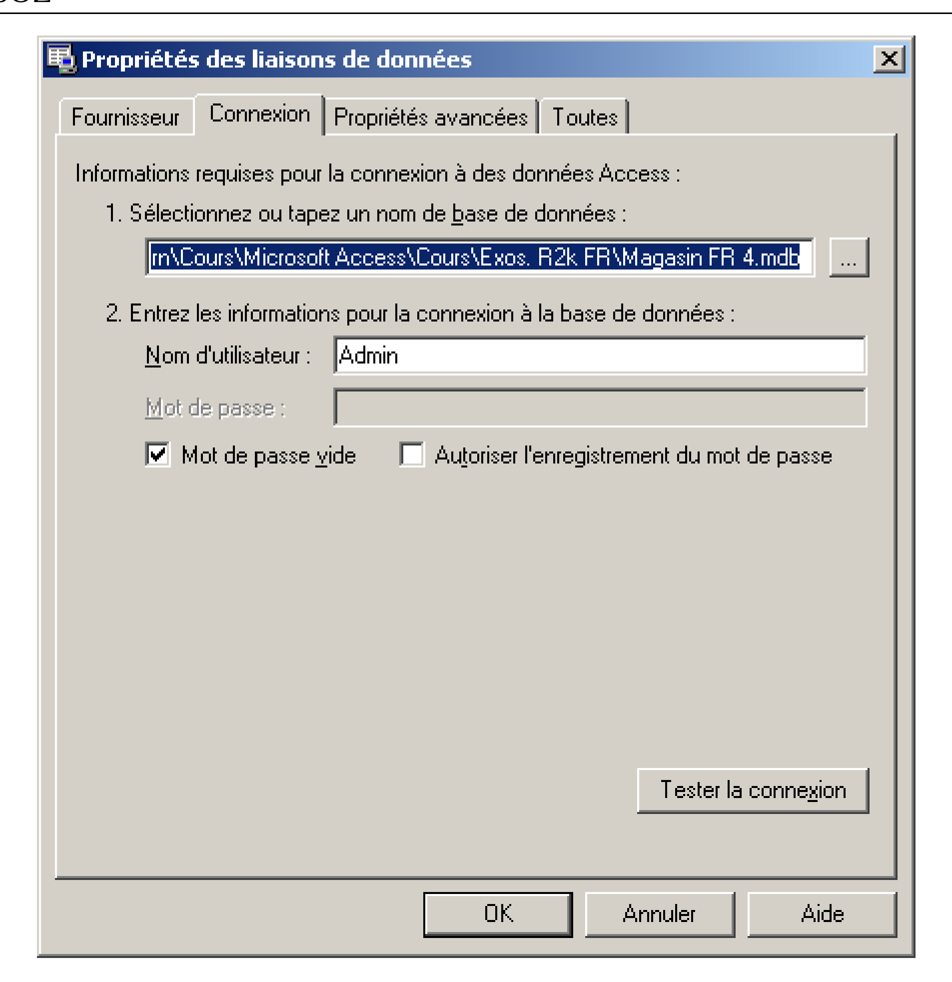


Si la fenêtre *Liste des champs* n'apparaît pas, allez dans le menu *Affichage.*

Une fois la page d'accès aux données formatée selon vos besoins et préférences cliquez sur l'habituelle disquette pour enregistrer la page au format HTML sur un disque réseau ou serveur IIS de votre entreprise.

Remarque: Evitez d'inclure des espaces et des accents dans le nom d'une page Web ou d'une page d'accès aux données.

Pour visualiser la liaison existante entre la page d'accès aux données et la base de données source, accédez à l'onglet *Base de Données* de la *Liste des champs*, cliquez avec le bouton droit sur le nom de la base puis cliquez sur l'option *Connexion*:



Attention! Si vous déplacez la base de données associée à la page, vous devez modifier le chemin d'accès à la base dans l'onglet *Connexion* de la boîte de dialogues *Propriétés des liaisons de données.*

Vous pouvez regrouper la page d'accès aux données en mode création. Pour cela, ajoutez si besoin est, le champ sur lequel le regroupement doit être effectué puis sélectionnez-le. Affichez la barre d'outils nommé "Web".

Cliquez sur le bouton puis vous verrez le champ se déplacer vers la gauche avec un nouveau bouton représentant une croix. Enregistrez les modifications, lancez l'aperçu et observez le fonctionnement.

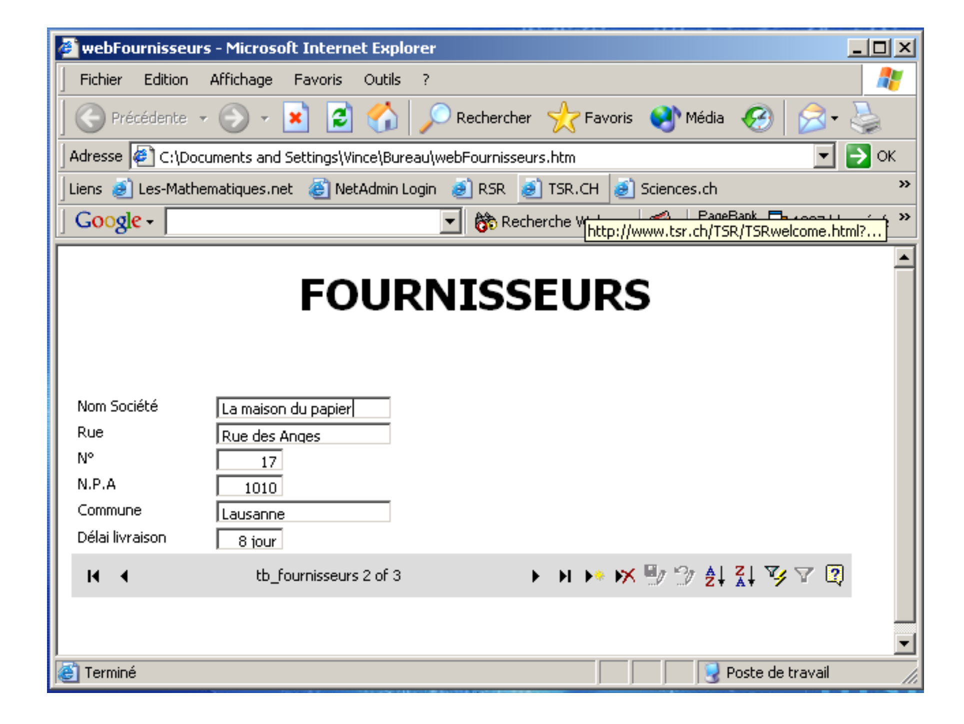
Evidemment, pour annuler le regroupement il vous faudra cliquer sur le bouton 

Si une boîte de dialogue apparaît vous indiquant que des paramètres de sécurité empêchent l'accès à une source de données située sur un autre domaine, activez l'option *Accès aux sources de données sur plusieurs domaines* dans Internet Explorer 5 (à peu près le même principe sur la version 6…).

Les composants Microsoft Office Web Component doivent être installés sur les postes clients et serveur.

Attention!!! Des données ne peuvent être saisies dans une page pour laquelle vous avez effectué des regroupements.

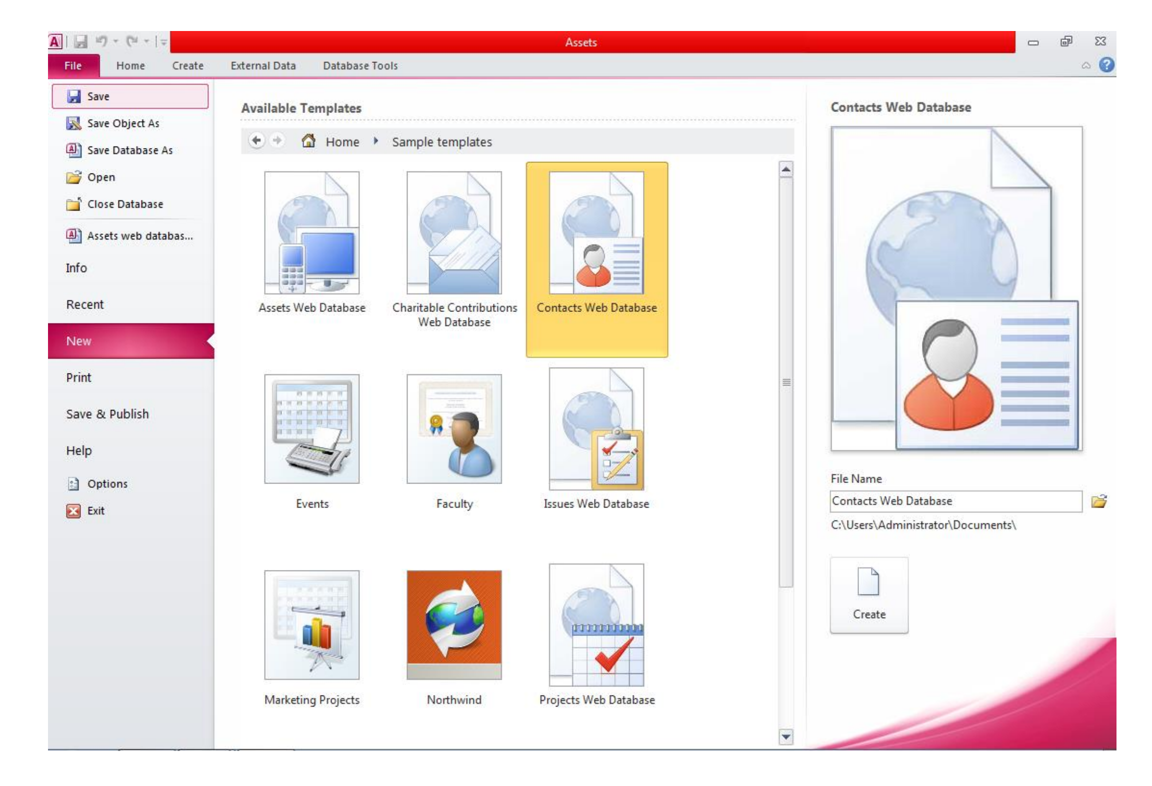
Voici à quoi ressemblera finalement la page web dans Internet Explorer 6.0:



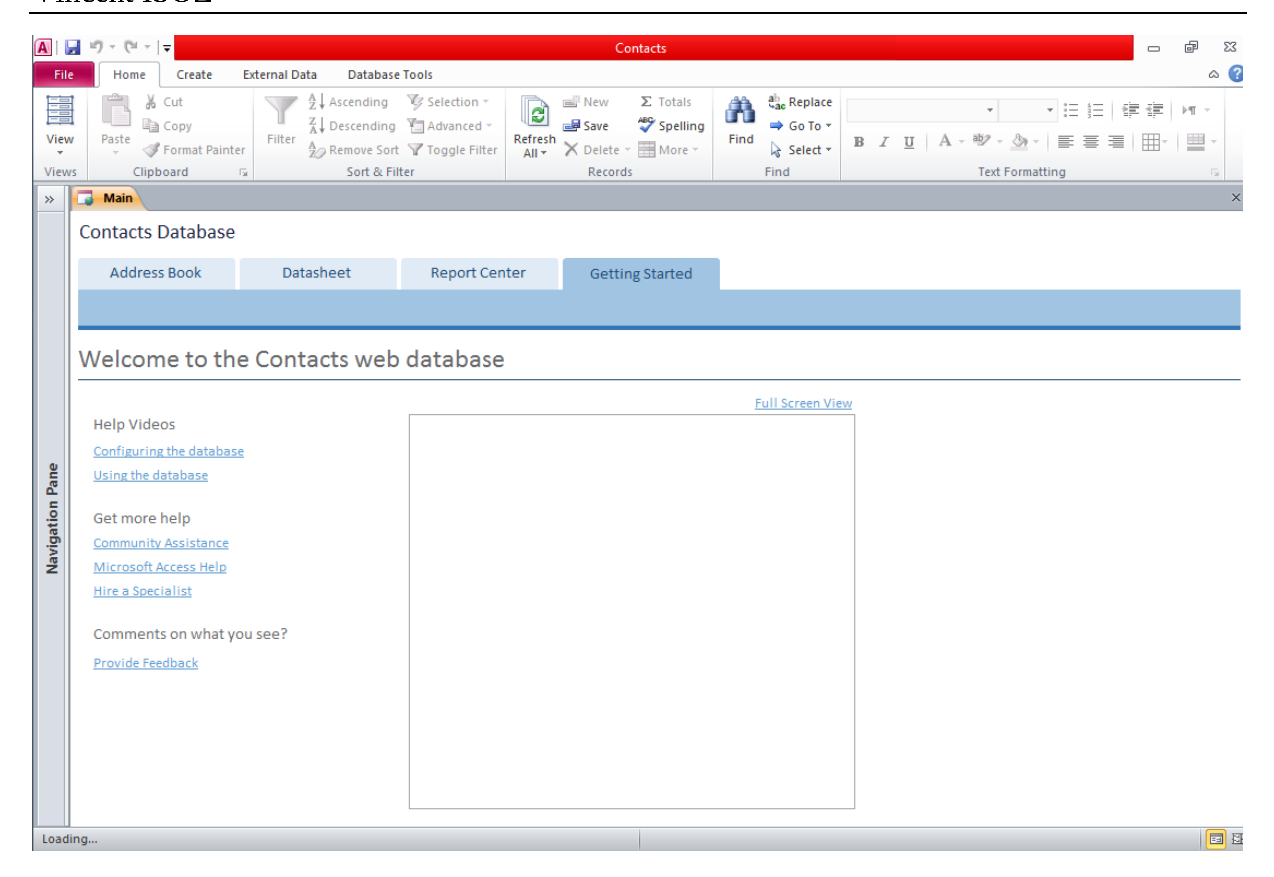
14.2 Application Web avec SharePoint 2010/Access 2010

Bon étant donné que c'est une nouveauté c'est hyper limité étant donné que c'est la version 1.0... il ne faut pas en espérer des miracles...

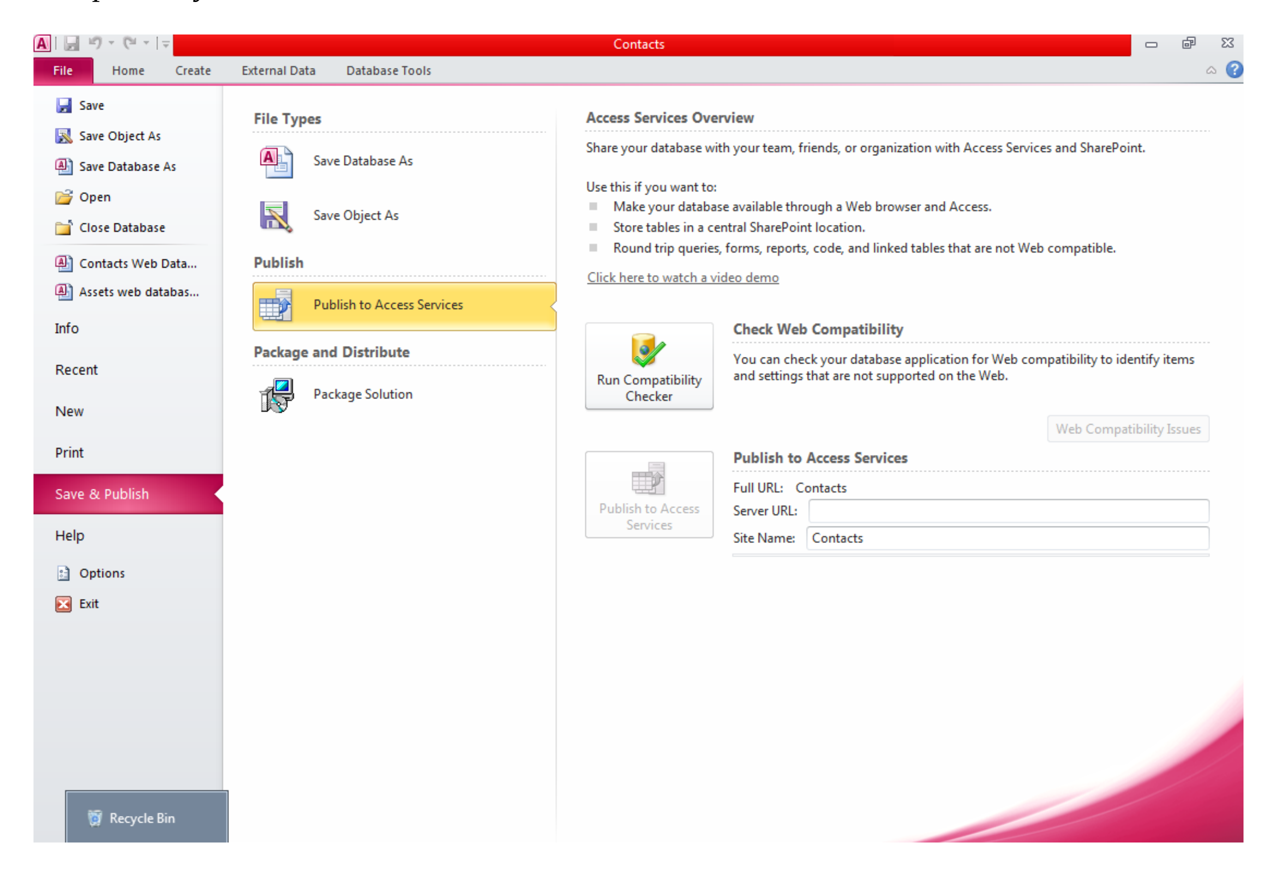
Bone première chose, nous ouvons Access 2010 et allons dans le menu *File/New* et nous prendront le traditionnel exemple de la base web *Contacts*:



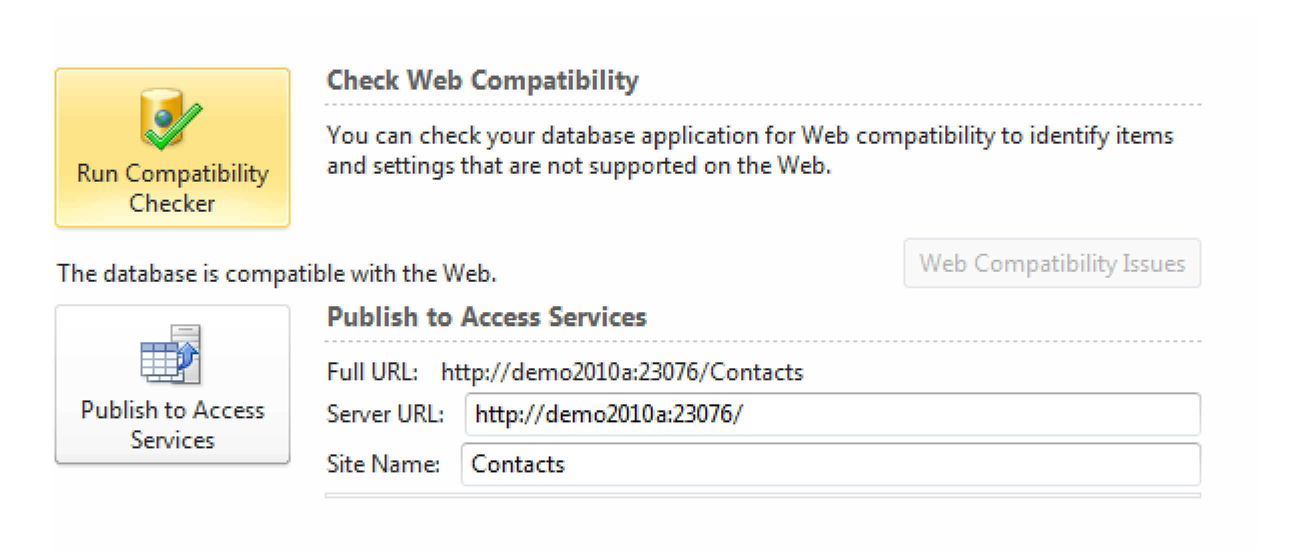
et vous cliquez sur *Create* Vous aurez alors:



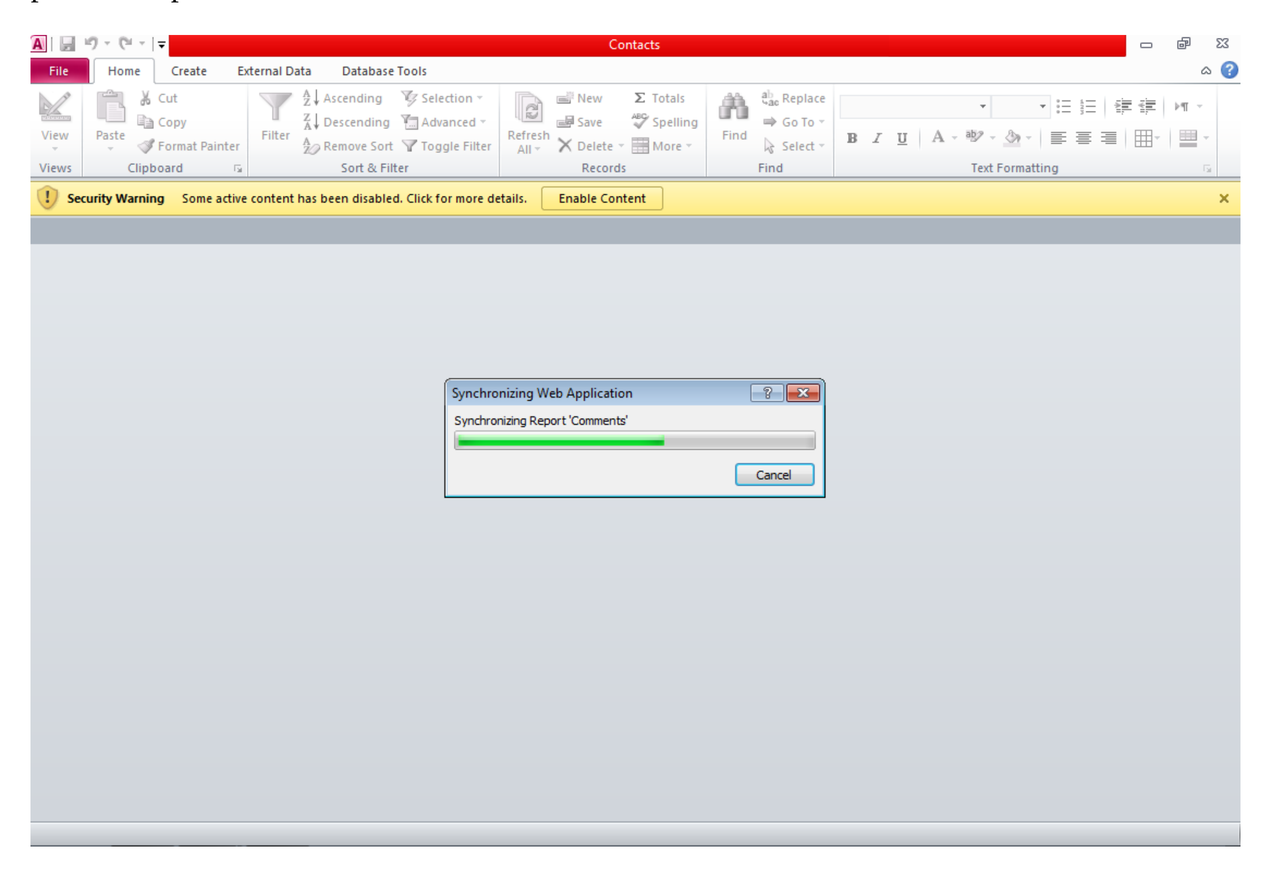
Ensuite, vous allez dans le menu *Save & Publish* où vous pourrez cliquer sur *Run Compatibility Checker*:



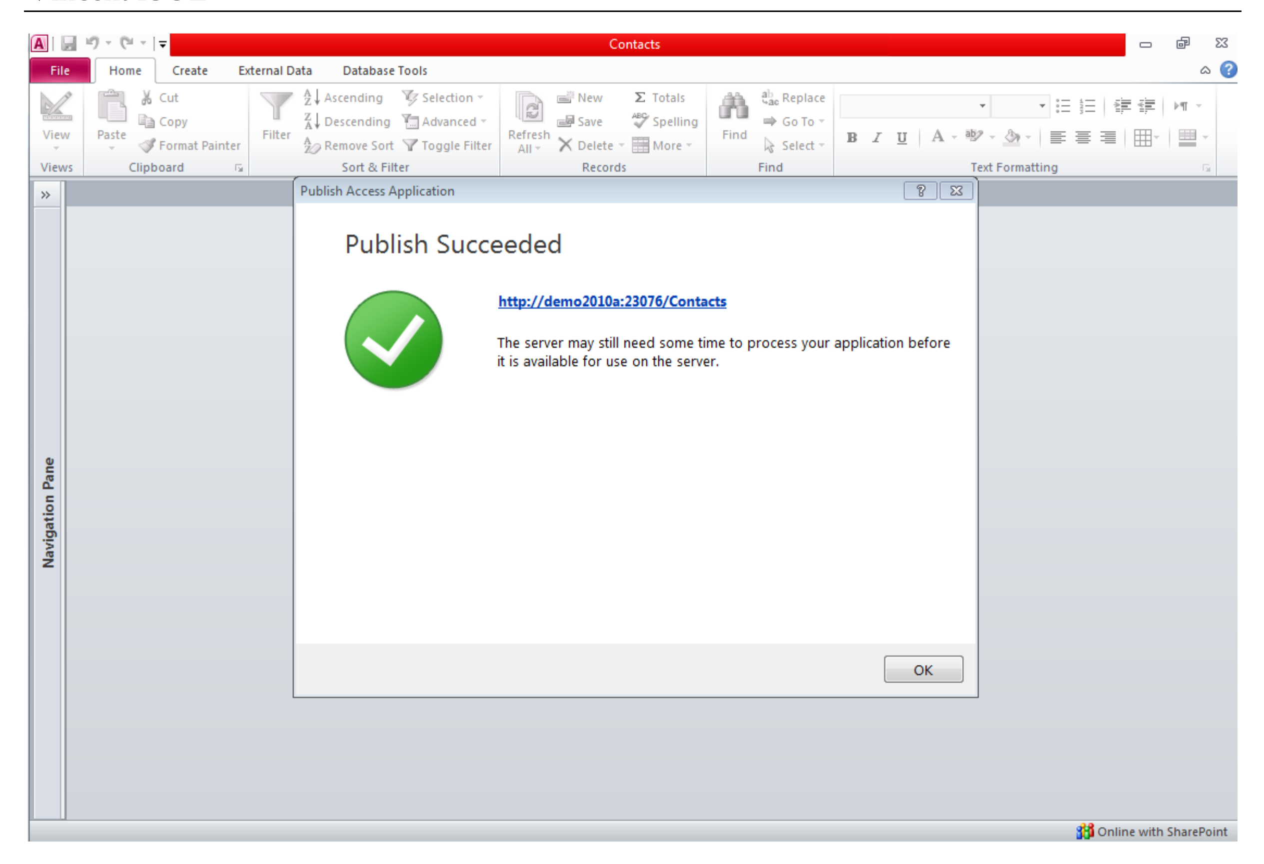
S'il n'y a pas d'erreurs, vous mettez l'adresse cible de votre site SharePoint. Attention à faire le bon choix car après pour déplacer la base c'est la joie...:



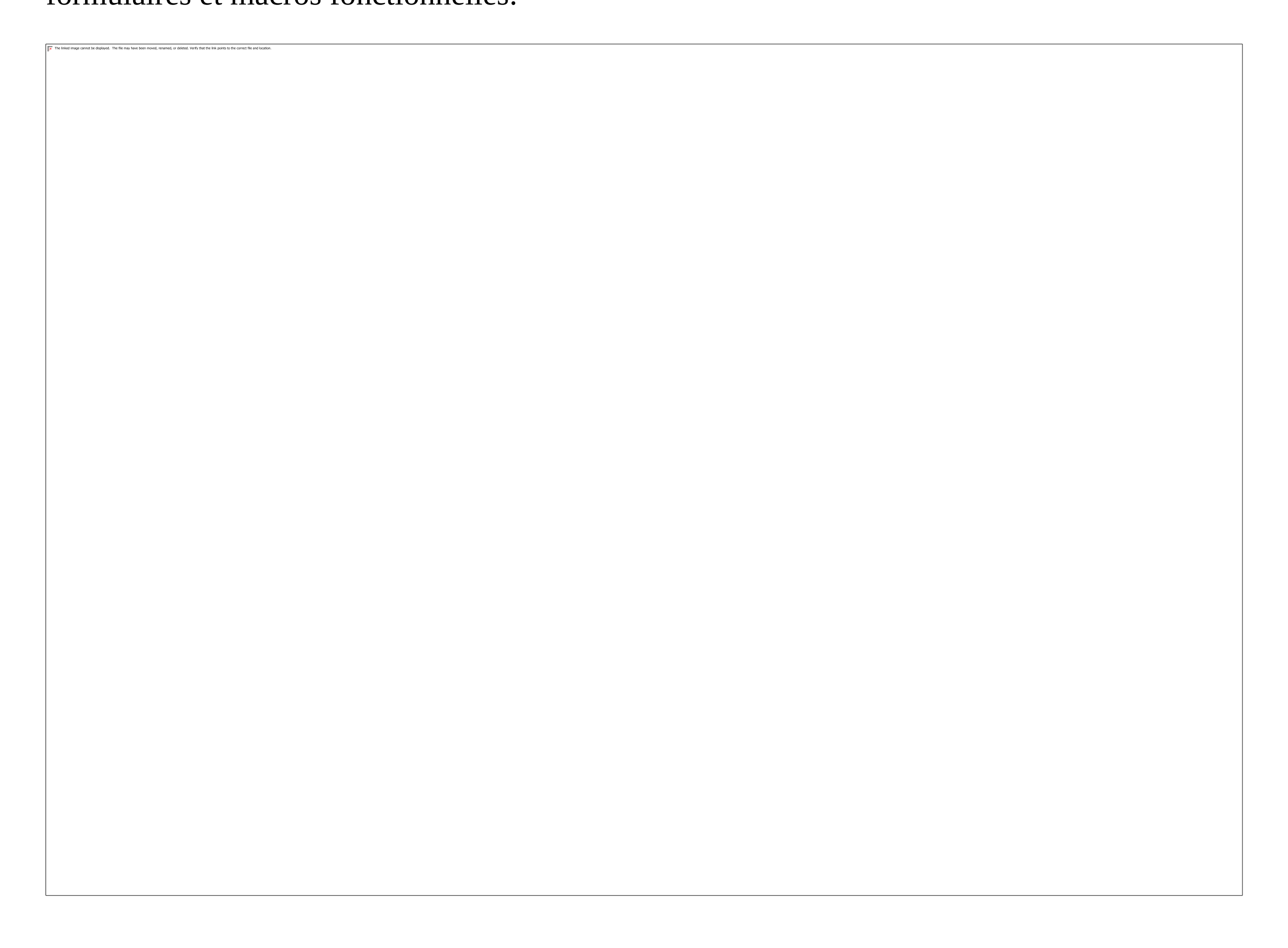
Vous validez en cliquant sur le bouton *Publish to Access Services* et vous devrez attendre un peu de temps:



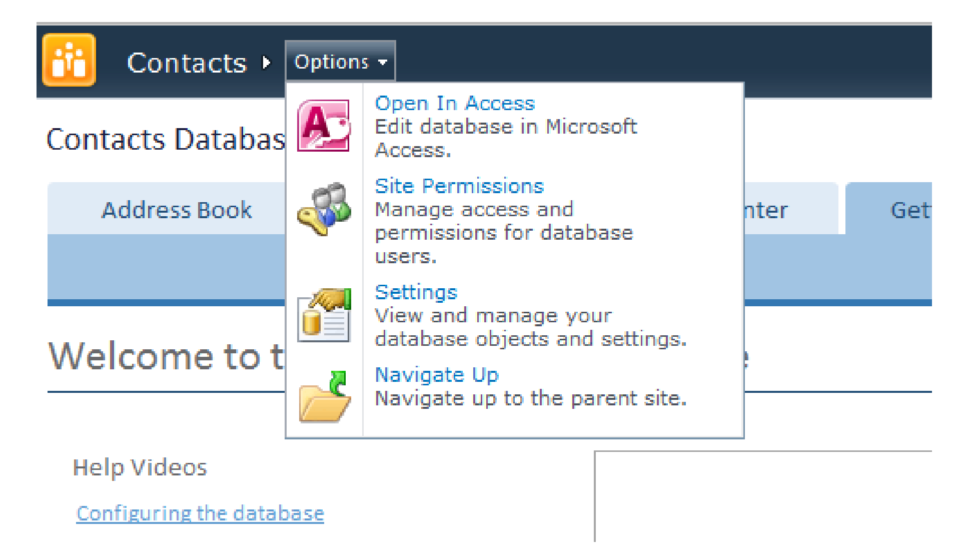
Une fois la conversion terminée sans erreurs vous aurez le message avec le lien direct vers le site SharePoint comme visible ci-dessous:



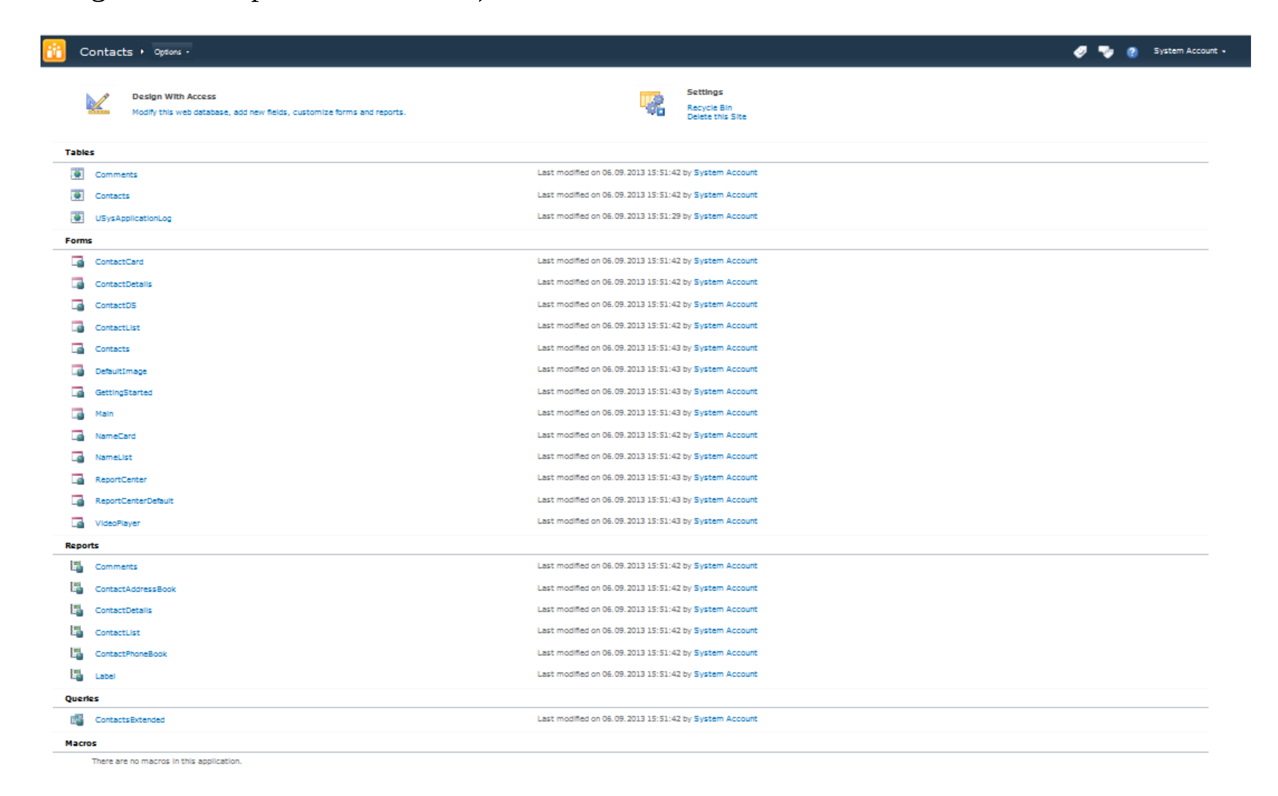
Quand vous cliquez sur le lien vous voyez votre base Access en mode web avec les requêtes, formulaires et macros fonctionnelles:



Efin remarquez les choix disponible dans le menu *Options*:



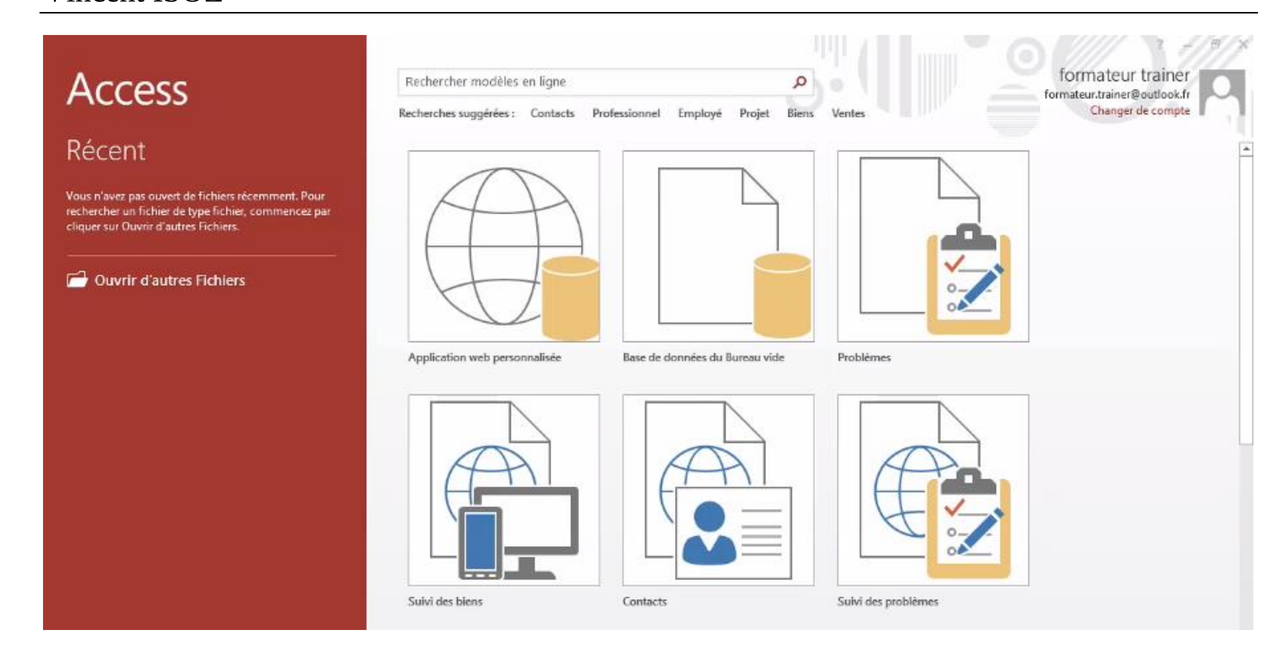
Si nous cliquons sur *Settings* nous voyons tout le contenu de la base avec la possibilité d'ouvrir dans SharePoint chaque élément (mais pour modifier il faut passer pour l'instant obligatoirement par le client lourd):



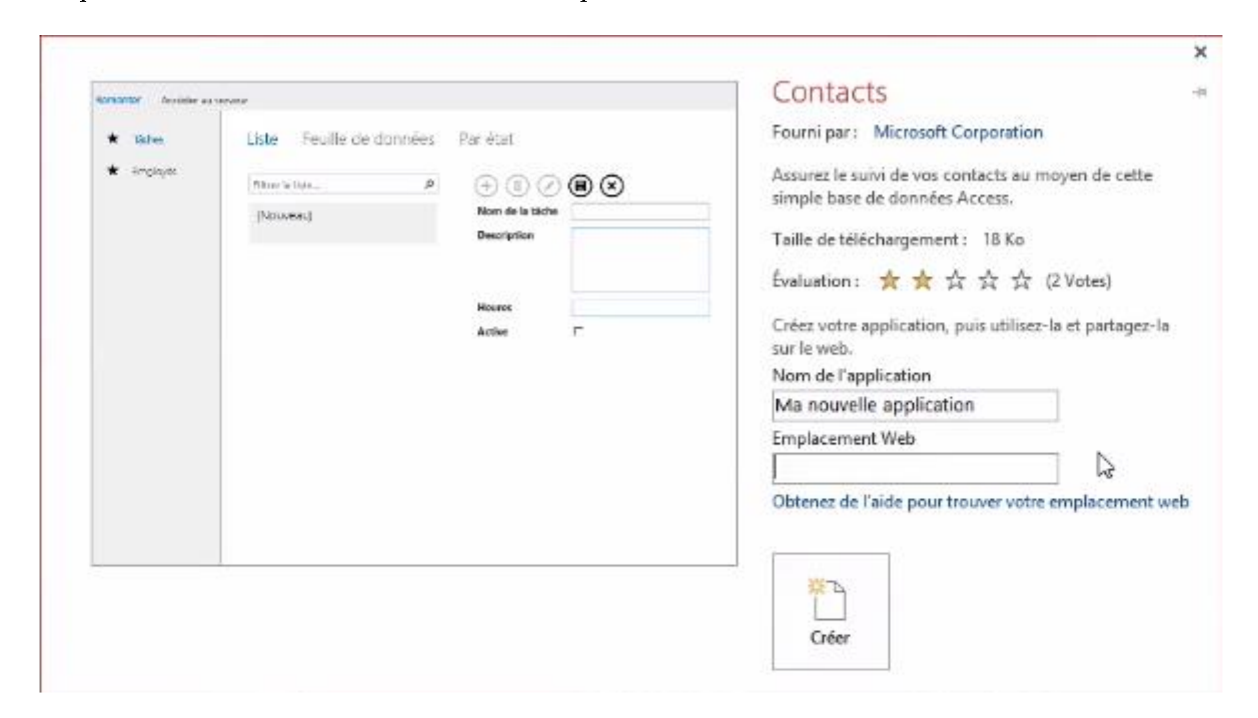
14.3 Application Web avec SharePoint 2013/Access 2013)

Bon cela reste limité mais il y a de nouvelles choses intéressantes par rapport à SharePoint 2010/Access 2010 avec cette version 2.0... Cependant, il ne faut toujours pas en espérer des miracles....

Nous repartons comme pour l'exemple SharePoint 2010/Access 2010 avec l'application web *Contacts*:

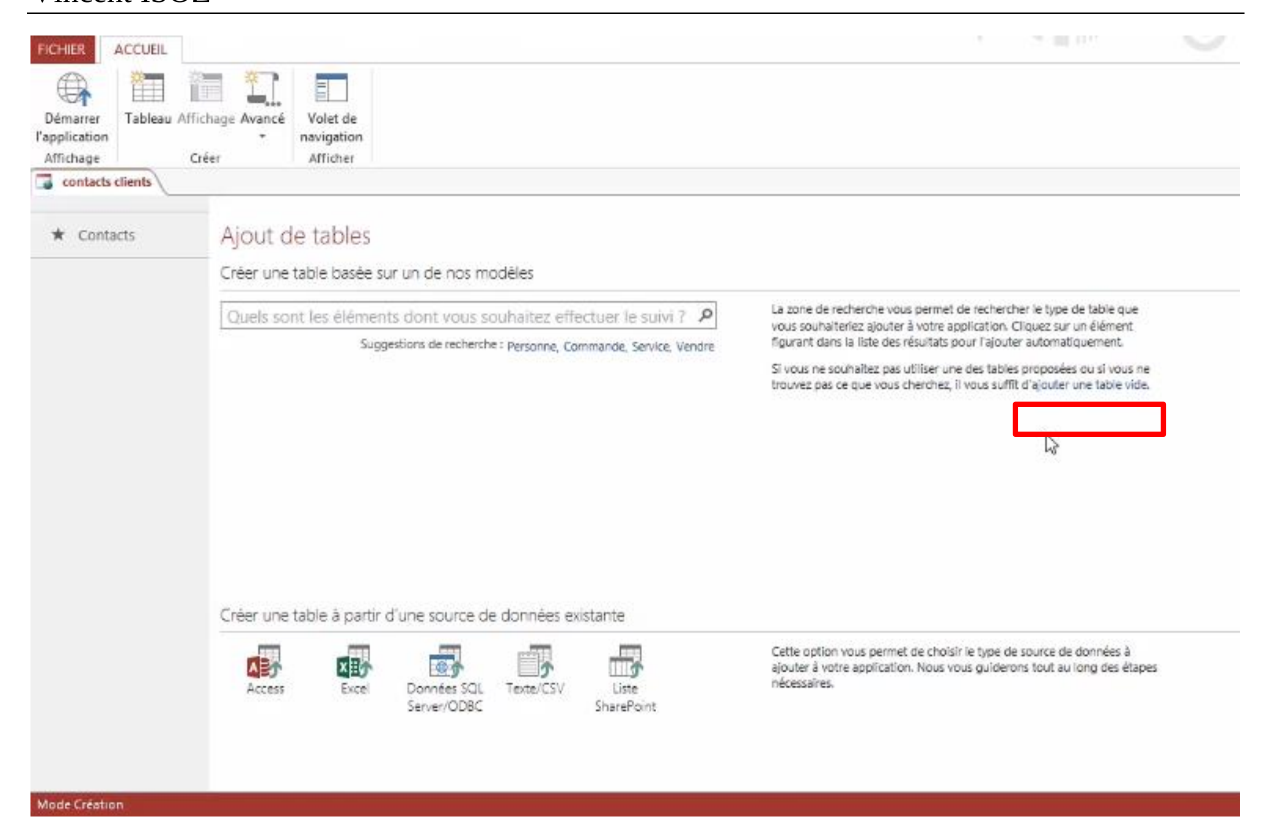


et que vous saisissez les informations adéquates:

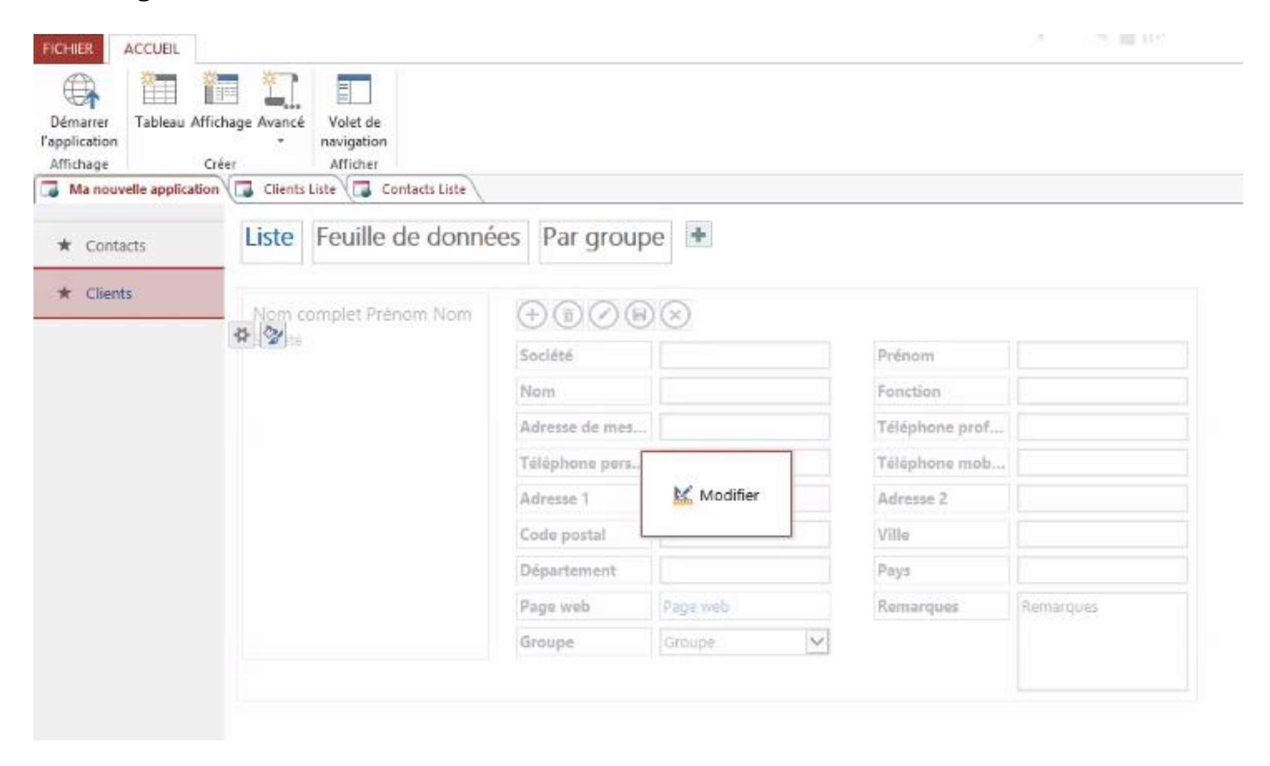


vous pouvez saisir l'URL de destination du site SharePoint de votre entreprise. Le lien sera créé dans la liste *Applications* du site cible.

Une fois la génération terminée, nous arrivons sur la vue suivante (remarquez le lien *Ajouter une table vide* mis en évidence ci-dessous):

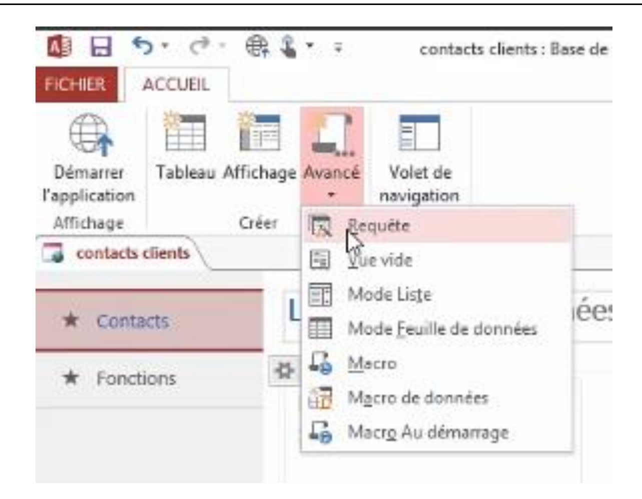


L'interface d'Access change du tout au tout avec uniquement (du moins pour Access 2013) deux onglets:



et le but n'est pas d'y faire de la saisie mais de modifier les formulaires (ce qui pour les modifications de base est trivial et similiare à ce que nous avons déjà vu)!

Remarquez le contenu du bouton *Avancé*:

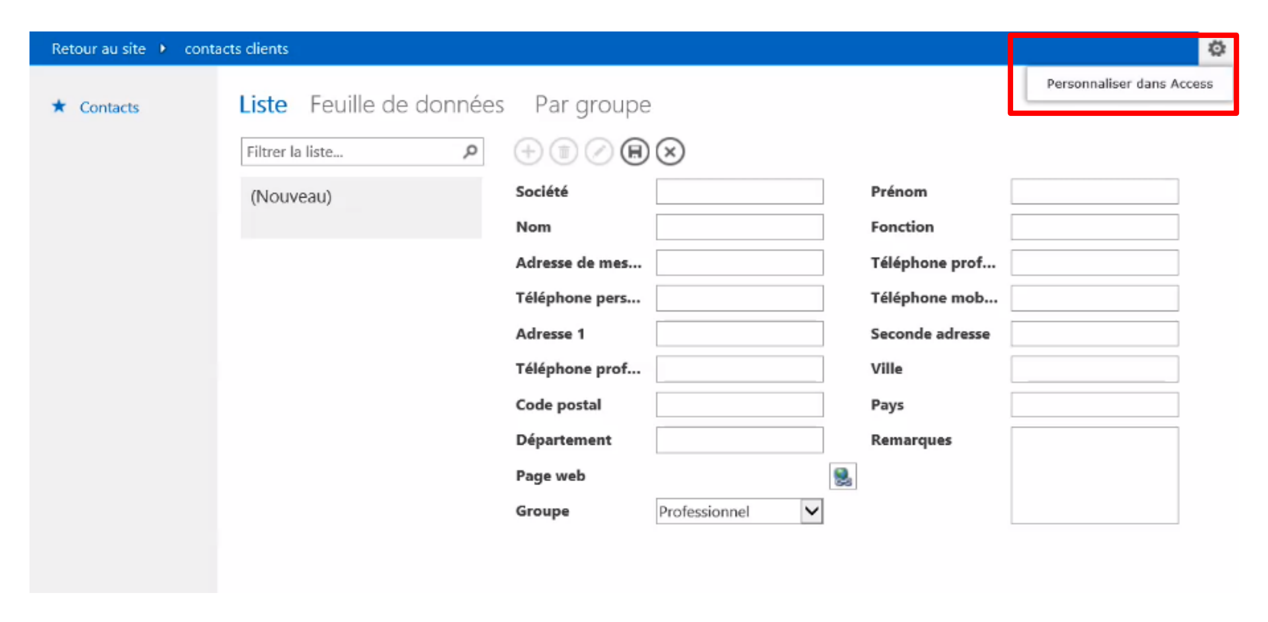


avec pour les requêtes les mêmes possibilité que les bases locales!!! (simplement les requêtes paramétrées ne peuvent pas être appelées depuis l'interface web...!)

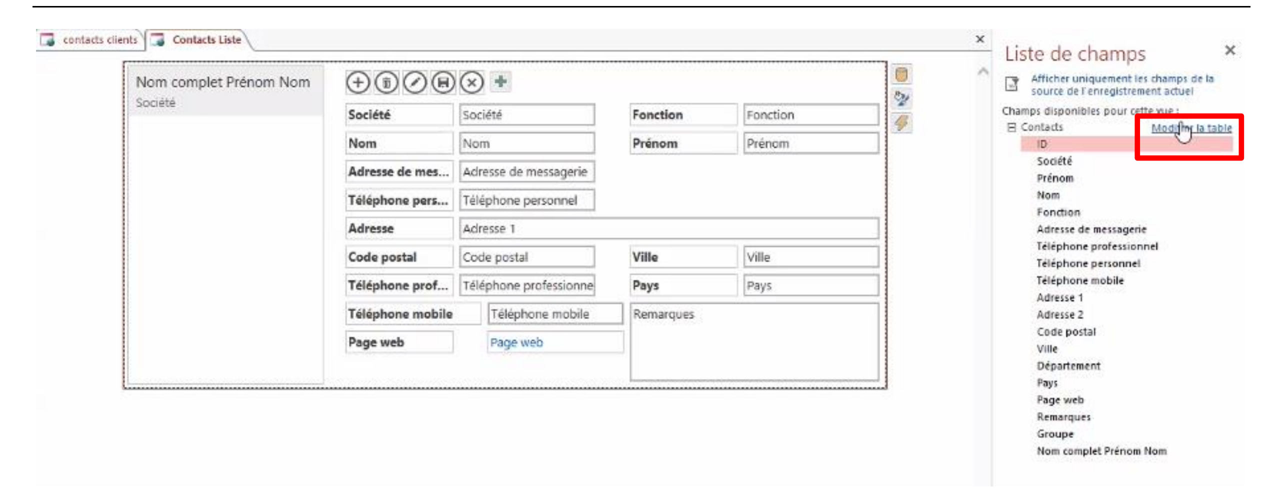
Si maintenant nous souhaitons voir ce que verront les utilisateurs, nous cliquerons sur:



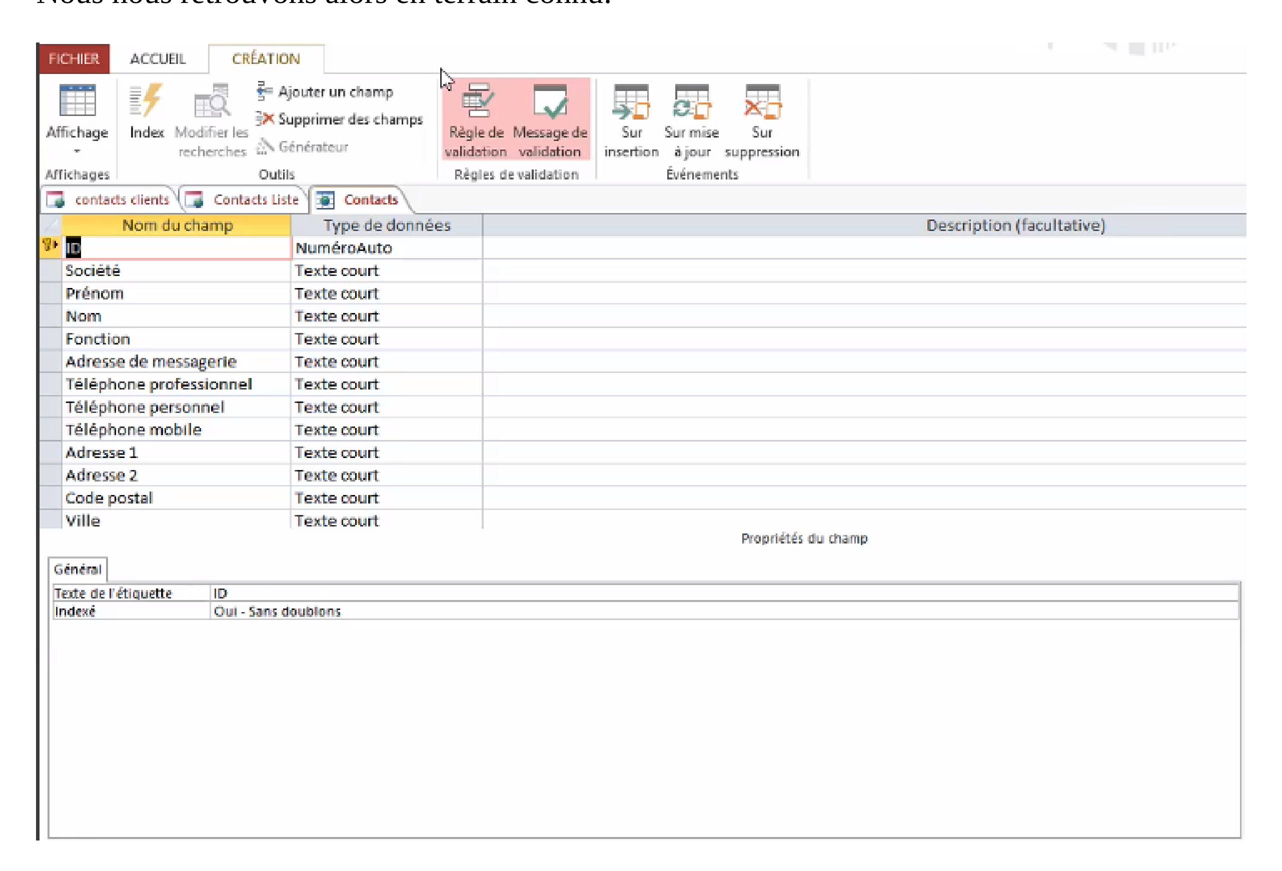
Nous arriverons alors sur la page SharePoint (remarquez la possibilité de re-modifier dans Access à tout moment via l'engrenage):



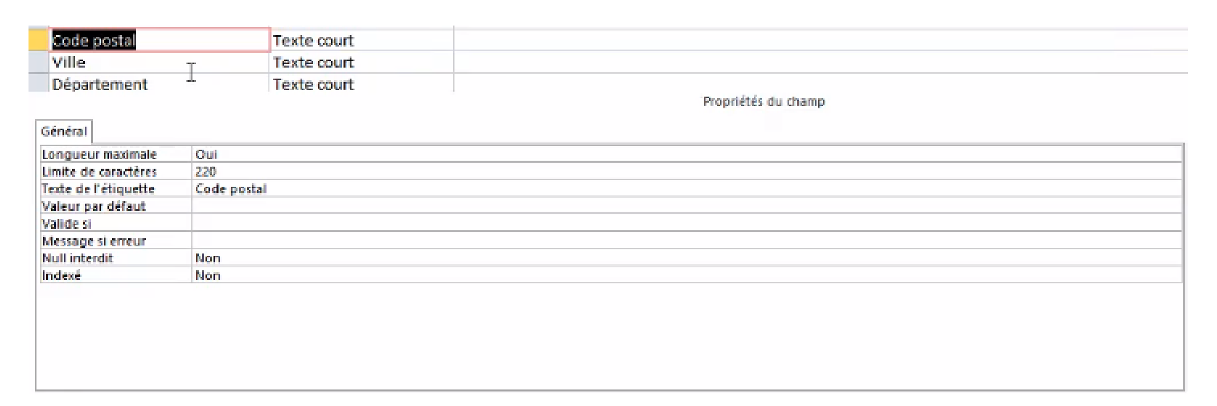
Maintenant, remarquez en mode modification le lien mis en évidence en rouge *Modifier la table*:



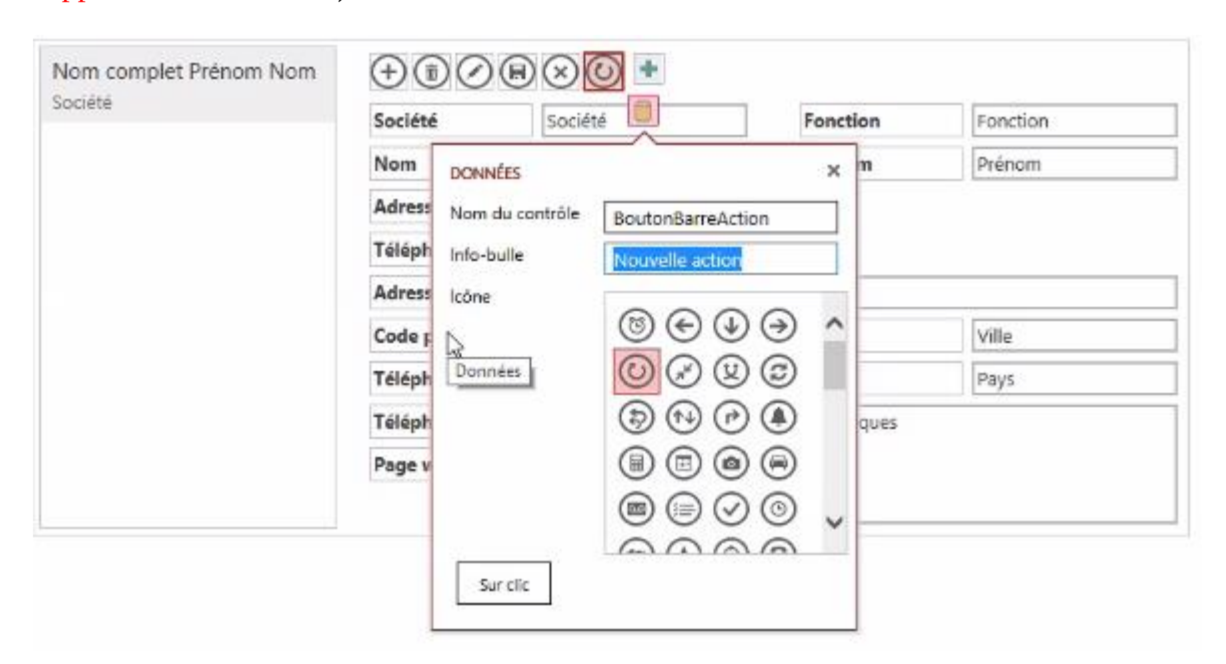
Nous nous retrouvons alors en terrain connu:



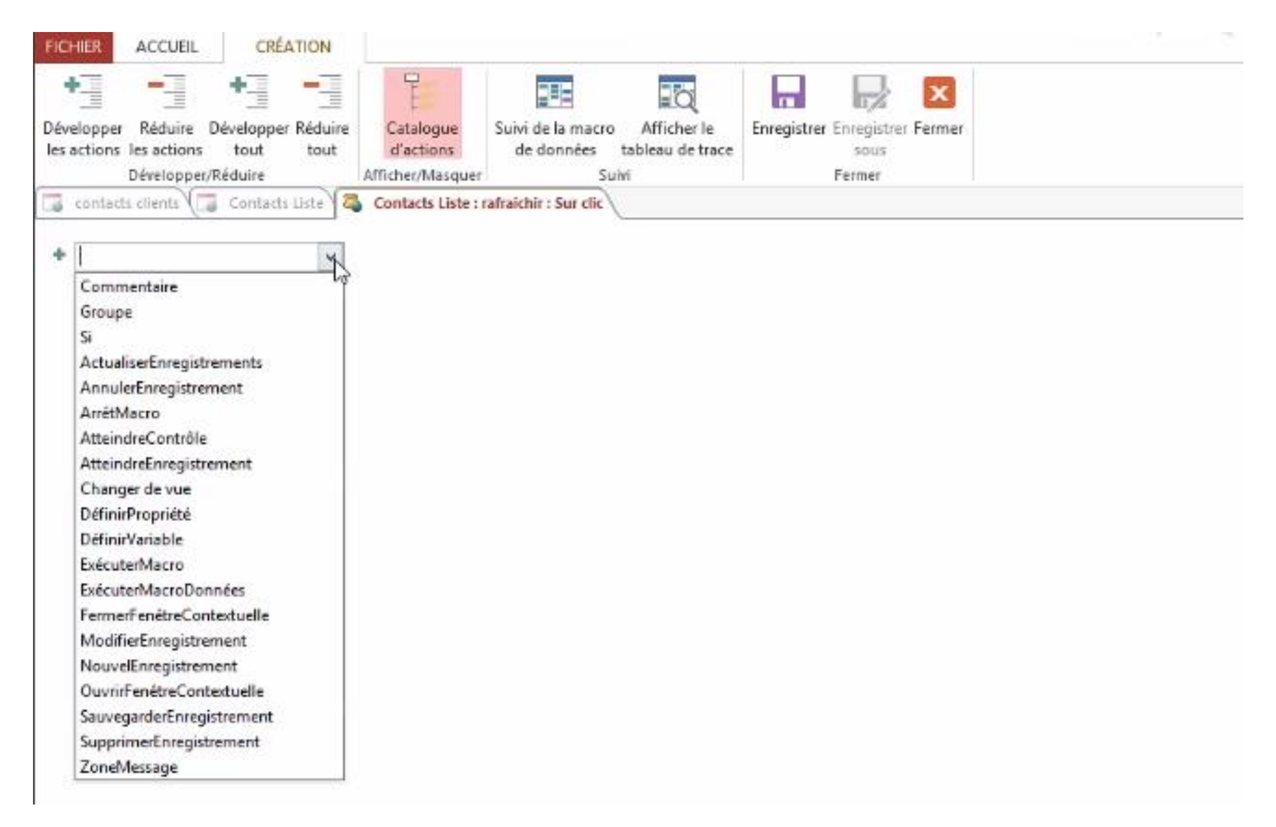
avec uniquement certaiens propriétés par rapport aux bases locales (remarquez l'absence des masques de saisie!! ce qui enlève presque tout intérêt professionnel à la l'outil):



et le lecteur aura très probablement remarqué que l'on peut ajouter des boutons (et aussi supprimer les existants!):

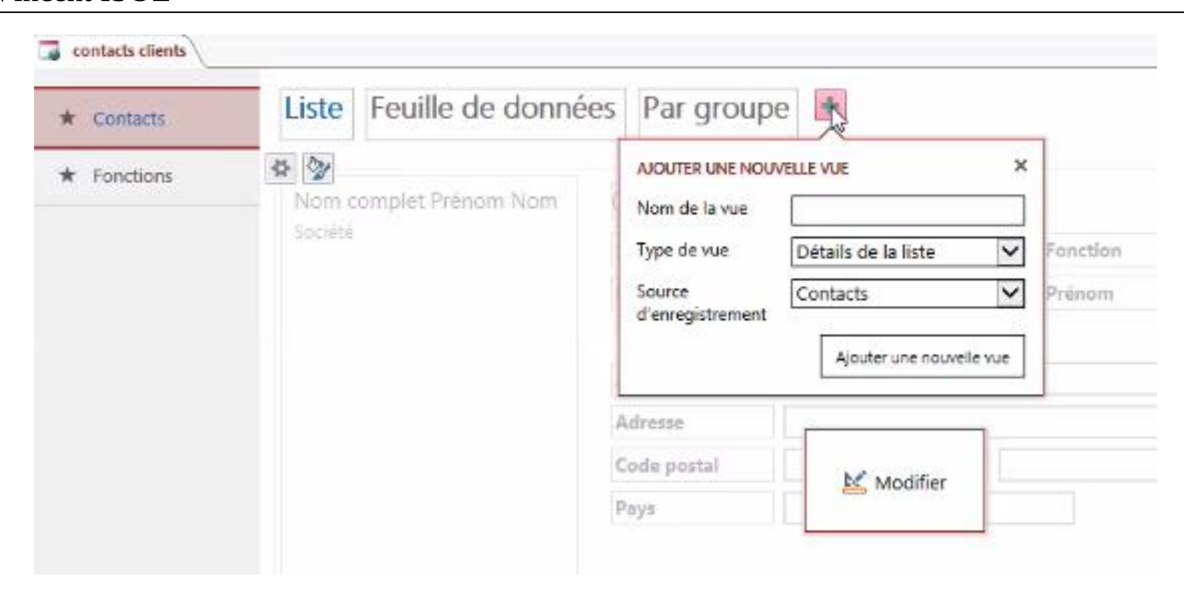


On choisira alors le nom du contrôle et l'info-bulle et lorsque l'on clique sur *Sur clic*, il vient l'assistant de génération de macros:



Donc rien de nouveau mais c'est quand même assez puissant lorsque l'on sait que cela va être transformé en script web.

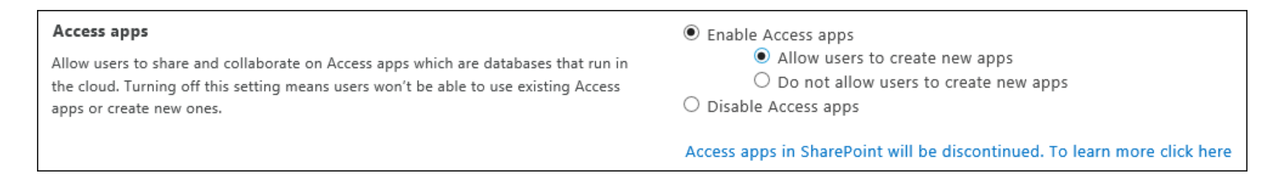
Il en va de même pour afficher des vues (d'autres tables ou requêtes non paramétrées):



14.4 Application Web avec Office 365

Donc d'abord attention!!! En Juillet 2018 Microsoft va désactiver Microsoft Access Services car ils n'en poursuivront pas le développement (annonce faire en Mars 2017).

Par contre pour ceux qui souhaiteraient toutefois faire du Microsoft Access, assurez-vous d'avoir les paramètres du serveur SharePoint de Office 365 l'option suivante activée sinon quoi vous ne pourrez rien faire:



Pour le reste… c'est la même chose, avec les mêmes limitations!

15 Optimisation et analyse

Une fois votre base terminée, il existe dans MS Access, une série d'assistants qui vont effectuer une analyse de votre base et qui vont tenter de vous aider à corriger les éventuelles erreurs ou manque de rigueur de votre part.

Nous aurions pu voir ce outil dès le début du cours mais il fallait au préalable certaines connaissances. Par ailleurs, lors de la discussion sur la normalisation des base de données dans le chapitre de modélisation, nous avons fait référence au présent chapitre.

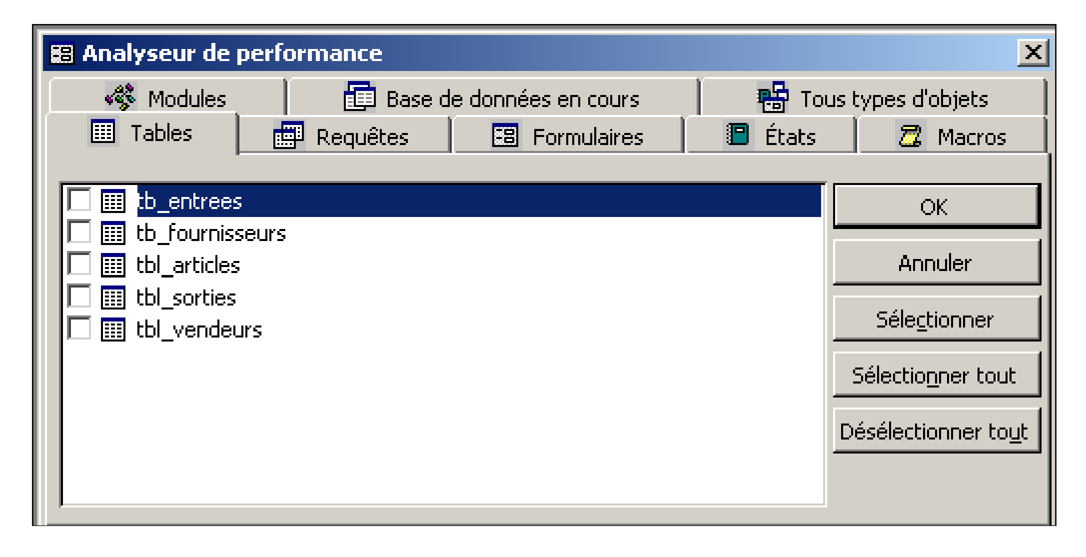
15.1 Table

L'outil *Table* me dans le menu *Outils/Analyse* est assez (voir très) médiocre. Il s'agit en fait d'un outil d'aide à la normalisation de deuxième et troisième forme mais qui ne peut deviner ce que vous avez en tête comme stratégie de développement.

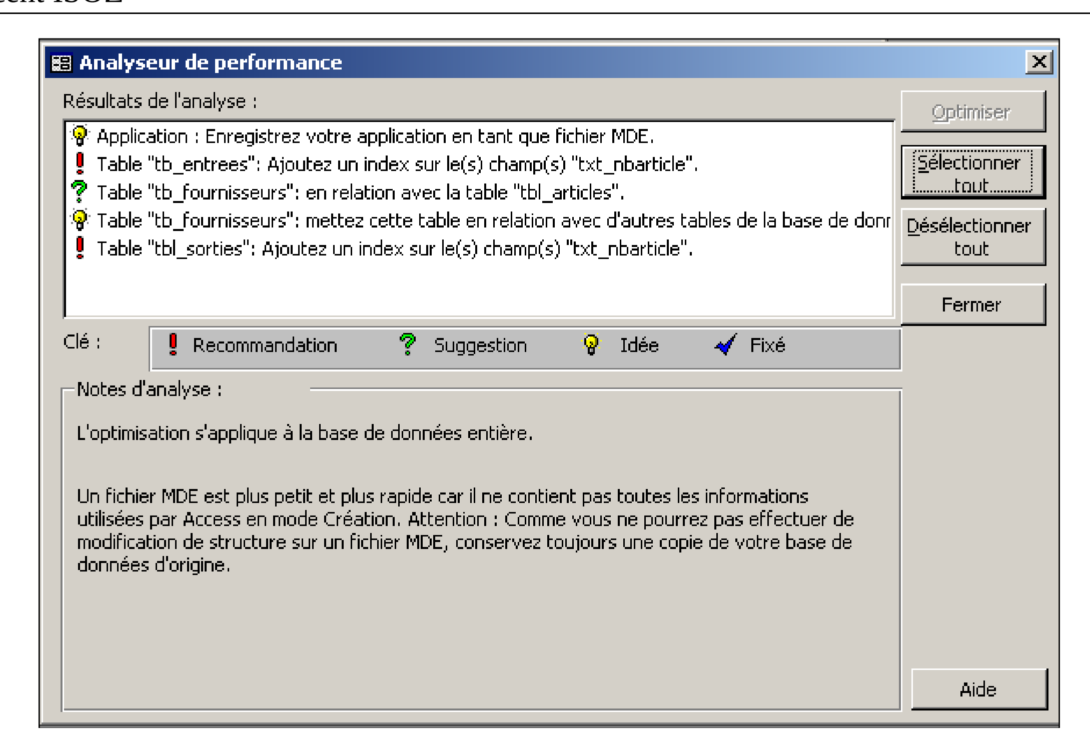
Pour exemple, prenons la table *tblSorties* dans une version complétement erronée (en 1ère forme normale) sans qu'elle soit liée à quoi que ce soit donc:

15.2 Performances

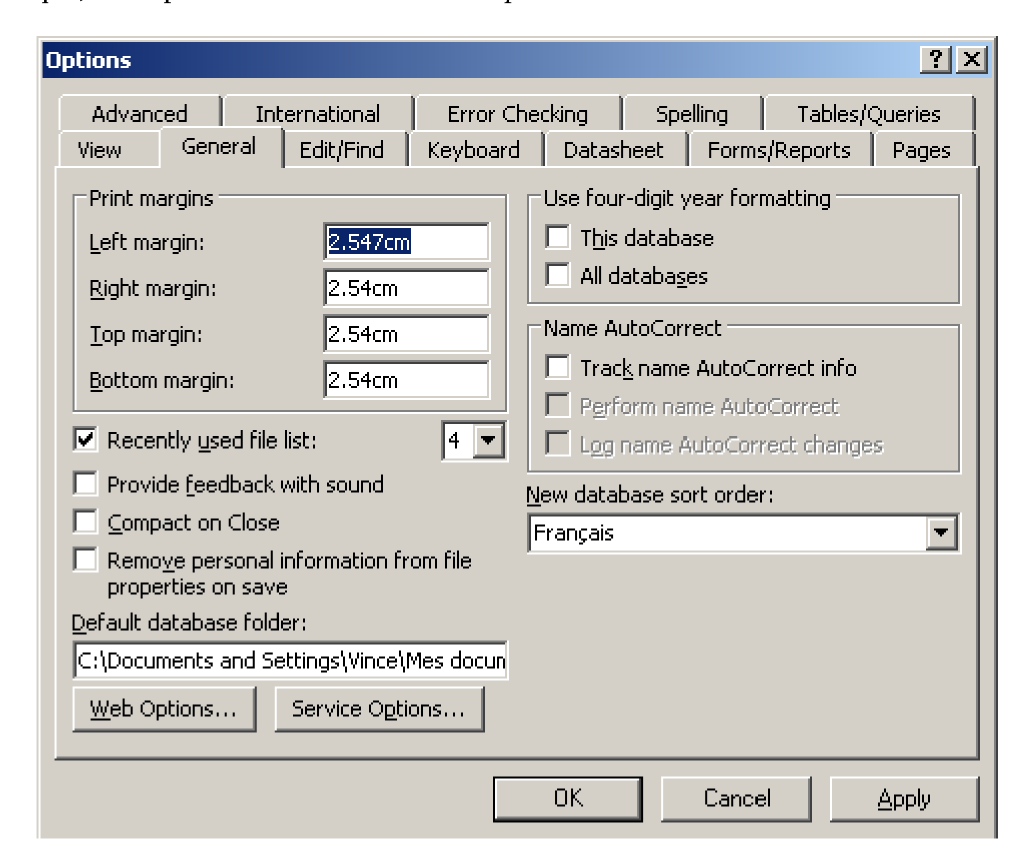
Dans *Outils/Analyse* lancez l'option *Performances*:



L'idéal, serait de ne plus avoir aucun message dans les différentes analyses. Ainsi, si vous sélectionnez l'onglet "Tous types d'objets" et cliquez ensuite sur le bouton "Sélectionner tout" et enfin sur le bouton "OK", vous pourriez être amené à avoir un résultat du genre:



Autre point important: Microsoft Access ne recouvre pas automatiquement l'espace libre après avoir effacer des enregistrement ou des objets. Pour recouvrir cet espace, vous devez compacter la base de données. Pour ce faire, allez dans le menu *Outils/Utilitaire de base de données* et sélectionnez l'options *Compacter la base de données*. Pour éviter à avoir faire ceci systématique, vous pouvez aller dans *Outils/Options:*



Vous remarquerez l'option *Compacter à la fermeture* dans l'onglet *General* qui vous évitera d'oublier cette procédure.

15.3 Documenter

16 Distribution

Votre application est terminée, et vous souhaitez maintenant la partager avec vos collègues ou autres utilisateurs (si vous êtes développeur). Dans le cas le plus simple une solution consiste à placer la base sur le disque réseau commun de l'entreprise afin que chaque utilisateur puisse l'exécuter sans complications. Cependant, avant de faire ceci, il est bon de penser à la manière dont vous allez la distribuer et faire de la maintenance.

Si votre ordinateur est connecté à un réseau, vous pouvez travailler dans une base de données Microsoft Access en même temps que d'autres utilisateurs. Vous pouvez partager des données dans un environnement multi-utilisateur de plusieurs manières.

1. Partager une base de données MS Access tout entière

   Placez la base de données MS Access entière sur un serveur de réseau ou dans un dossier partagé. Cette méthode est la plus facile à mettre en œuvre. Chaque utilisateur partage les données et utilise les mêmes formulaires, états, requêtes, macros et modules. Utilisez cette stratégie si vous souhaitez mettre en place une utilisation uniforme de la base de données ou si vous ne voulez pas que les utilisateurs puissent créer leurs propres objets.

2. Ne partager que les tables de la base de données MS Access

   Placez uniquement les tables (la base "Jet") sur le serveur du réseau et conservez les objets de la base de données dans les ordinateurs des utilisateurs. Ainsi, les performances de la base de données sont accrues, car seules les données sont transférées sur le réseau. De plus, les utilisateurs peuvent personnaliser leurs formulaires, états et autres objets en fonction de leurs préférences et besoins individuels, sans affecter les autres utilisateurs.

   Vous pouvez séparer les tables des autres objets de base de données à l'aide de l'outil *Fractionner la base de donnée*s (Outils/Utilitaire de base de données).

   Au besoin, on pourra remplacer la base Jet par une base MSDE (Microsoft SQL Server Desktop Edition) ou client/serveur.

   Jet et MSE sont limités en taille: 2Go maximum. Au-delà, il faudra utiliser une base client/serveur proprement dite.

   Jet, avec une limite théorique de 255 connexions simultanées, marque une baisse de performances à partir de 5 connexions, et n'est pas recommandé au-delà de 10. MSDE est optimisé pour 5 requêtes concomitantes, y compris les procédures stockées. Au-delà, les performances se dégradent rapidement.

3. Partager la base de données MS Access sur Internet

   Vous pouvez convertir un ou plusieurs objets de base de données au format HTML statique ou HTML généré par serveur, ou créer des pages d'accès aux données, puis les afficher dans un navigateur tel que Internet Explorer, sur le Web

4. Répliquer la base de données MS Access

   Si vous utilisez deux ordinateurs, un ordinateur de bureau et un portable par exemple, vous pouvez utiliser le Porte-documents de Microsoft Windows pour effectuer des réplicas de votre base de données Access et les maintenir synchronisés. En outre, plusieurs utilisateurs situés à divers endroits peuvent travailler simultanément avec leurs propres copies et les synchroniser à travers le réseau, par l'intermédiaire d'une connexion par numérotation téléphonique ou via Internet.

5. Créer une application client-serveur

   Si vous travaillez dans un environnement client-serveur, vous pouvez tirer profit de la puissance et de la sécurité supplémentaires qui vous sont offertes en créant une application client-serveur. Pour plus d'informations, il faudrait suivre un cours Visual Basic et MS SQL Server ou ASP/PHP.

Voici à toutes fins utiles à un tableau récapitulatif:

Tableau 6 Stratégies de déploiement

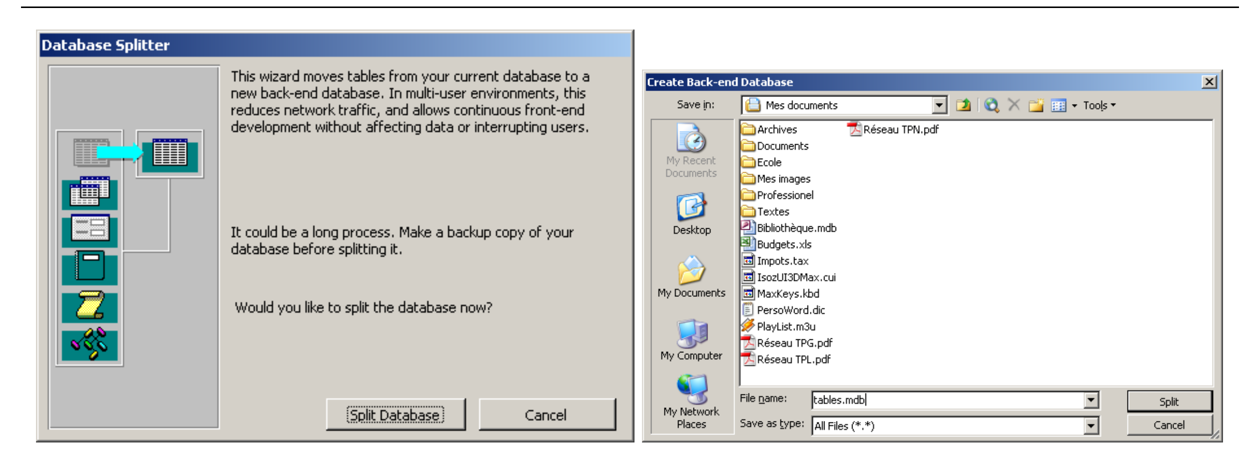
Réseau	Application	Type de Base	Contraintes
	Mono Poste	Jet (.mdb)	Convivialité maximale. Performances maximale
Réseau local peu encombré	Base partagée 5 à 10 utilisateur maxi.	Jet	Si pas trop de conflits sur les blocages convivialité maximale. Performances acceptables
		MSDE	Perte de souplesse. Bonnes performances
Réseau local avec serveurs (de domaine, de fichiers, de mail)	Distribuée, de un à plusieurs dizaines d'utilisateurs, mise à jour périodique	Jet en tampon: une copie (ou un extrait) de la base sur chaque pose, synchronisée avec la base client/serveur ou Jet répliquée	Convivialité maximale. Performances maximales. Flexibilité de la base tampon.
	Plusieurs centaines, voire milliers d'utilisateurs, mise à jour immédiate de la base centrale	Client/serveur	Convivialité maximale. Bonnes performances.
Internet	Base en lecture seule	Jet zippée, en téléchargement	Convivialité maximale. Performances maximales.
	Base moyenne en lecture/écriture	Jet répliquée	Convivialité maximale. Performances maximales. Réplication à mettre en place.
	Bases lourdes, plusieurs milliers d'utilisateurs.	Client/serveur, avec ou sans réplication	Convivialité Internet. Performances Internet.

16.1 Fractionnement d'une base

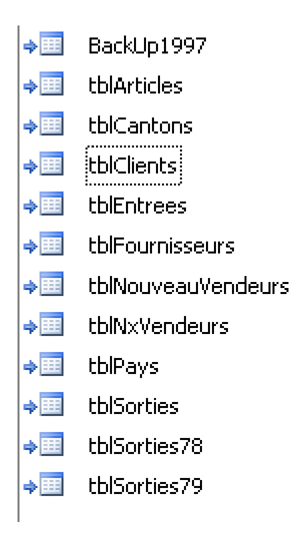
Objectif: Ne partager que les tables de la base de données MS Access

Allez dans le menu: *Outils/Utilitaire de base de données/Fractionner la base de donnée*s.

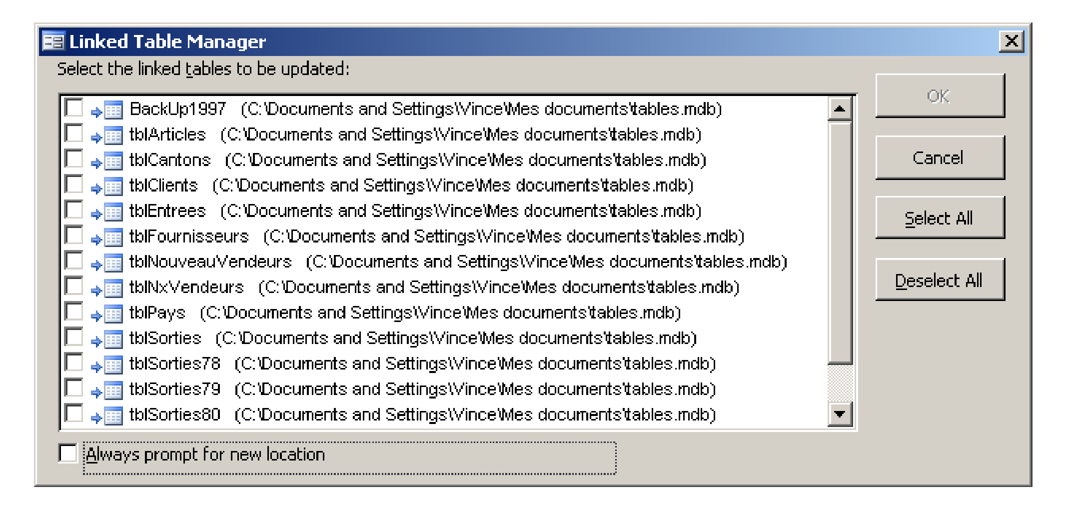
Suivez l'assistant et renommez le fichier contenant les tables ainsi: *Tables.mdb*



Vous verrez dans la fenêtre base de données que chaque table comporte maintenant une flèche ce qui signifie qu'il y a une liaisons externe.



Au cas où la liaison serait perdue ou modifiée, il faut aller dans le menu *Outils/Utilitaire de base de données/Gestionnaire de tables liées*:



Il suffit alors de cliquer sur *Select All* et ensuite sur *OK* et de définir le nouveau chemin du fichier mdb contenant les tables.

Nous allons placer les tables de la base "la plus propre" de la classe sur le serveur (ou de chaque participant respectif)

Distribuons le fichier contenant les formulaires, requêtes et états… sur chaque poste et effectuons-y la liaison.

Vérifiez le fonctionnement.

Remarque: dans une entreprise il est malheureusement fréquent que groupe de travail ait développé sa propre base. Quand vient le temps pour des raisons de cohérence et de productivité de réunir toutes les bases il existe une possibilités qui revient à lier une table d'une base donnée à une autre base. Pour ce faire, la manipulation est trivialement simple, il suffit d'aller dans le menu *Fichier/Données externes/Lier une table* (nous avons déjà pratiqué cela au début du cours).

Nous verrons dans la partie VBA du cours comment lorsque nous sommes en possession d'un fichier MDE, automatiser la mise-à-jour des liaisons.

16.2 Réseau

Lorsque vous travaillez en réseau avec plusieurs personnes:

1. Microsoft Access a besoin de beaucoup de bande passante sur le réseau. À moins de 100 Mb/s, ne faites pas d'applications partagées!

2. Privilégiez le runtime plutôt que le client lourd du développeur

3. Privilégiez l'utilisation des *.mde plutôt que des *.mdb ou *.accdb

4. Évitez de mettre l'application comme le fichier des données dans des sous-sous-sous répertoires et évitez les noms longs pour les répertoires et le fichier de données. Plus vous êtes proche de la racine et moins vos noms de répertoire sont courts (convention 8.3), plus les performances seront accrues.

5. Au moins une fois par demi-journée demandez à tout le monde de fermer complétement la base de données (sans la faire planter par le gestionnaire de tâches ce qui n'équivaut pas à la fermer!)

6. Si votre fichier de données est hébergé sur un serveur Windows (sauf Windows 2003 Server):

    a. Désactivez le délai de notification de violation de partage.

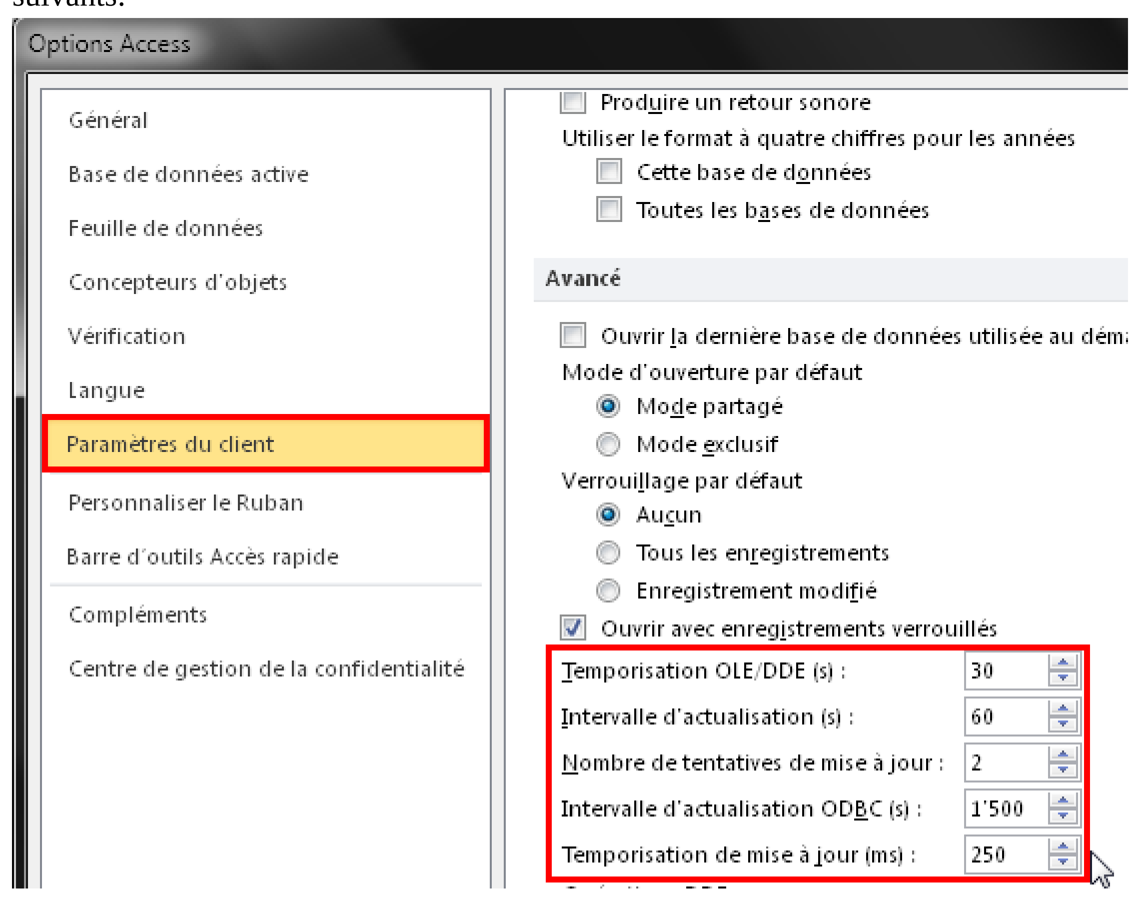
    b. Désactivez la génération automatique de nom de fichier court.

    c. Activez la mise en cache des noms longs (sauf si vous utilisez des noms courts).

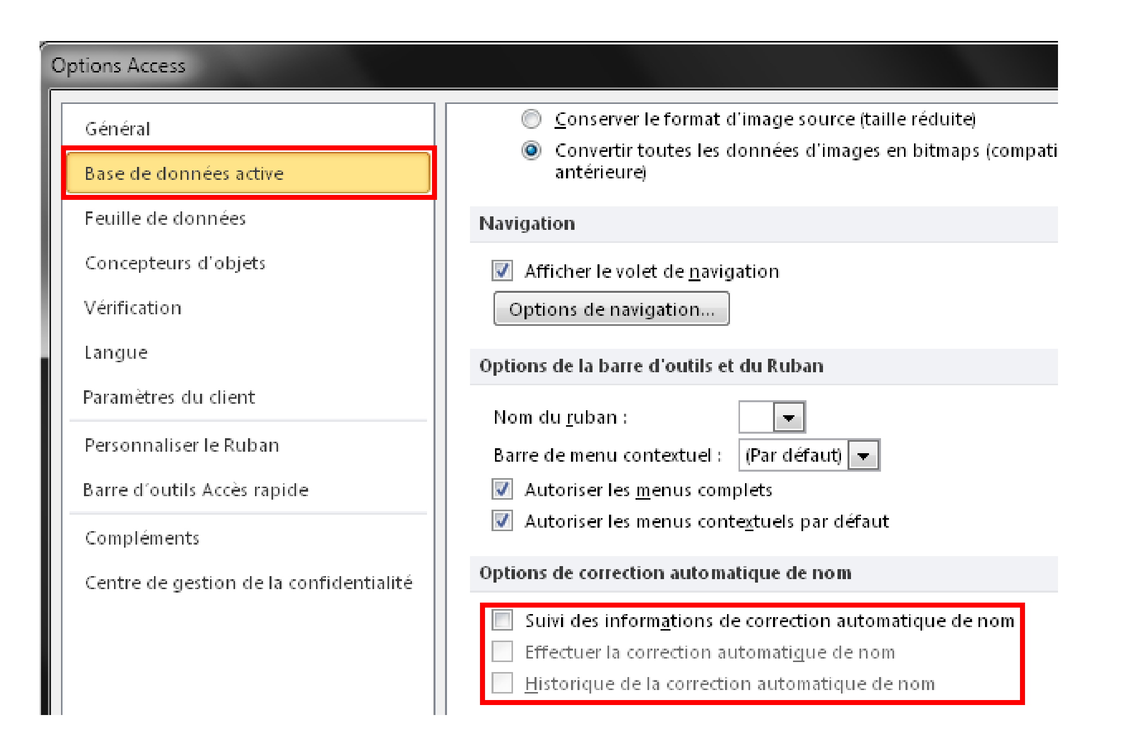
7. Certains Anti-virus trop agressifs nuisent à la bonne marche de nos machines, les applications en font les frais. N'hésitez pas exclure de la vérification les applications Microsoft Access.

8. Placez vos bases partagées sur des volumes NTFS plutôt que FAT ou FAT32.

9. Enfin dans le dernier des cas en allant dans *Fichier/Options/Paramètres du client* (versions 2007/2010/2013) vous pouvez jouer par tatonnement avec les paramètres suivants:



10. Ou encore en allant dans la section *Base de données active* vous pouvez décocher les éléments mis en évidence ci-dessous:



17 Securité avancée

Microsoft Office Access offre trois méthodes de protection d'une base de données:

1. Définir un **mot de passe** pour ouvrir une base de données…

2. Enregistrer la base de données dans un **fichier au format MDE** (déjà vu dans la théorie des macros)**,** ce qui supprime le code Visual Basic modifiable et empêche la modification de la structure des formulaires, des états et des modules.

3. Utiliser une **sécurité au niveau utilisateur**, ce qui permet de délimiter les parties auxquelles l'utilisateur peut accéder ou celles qu'il peut modifier.

Remarque: Désolé si ce chapitre n'est pas très clair mais à force que Microsoft change les versions à un rythme endiablé il y a un mélange de plus en plus grand dans le présent document…. évidemment tout ce qui est écrit ci-dessous est décanté en formation de manière correcte et propre (du moins j'essaie…).

17.1 Définition d'un mot de passe

La méthode la plus simple pour la (pseudo-)sécurité est de définir un mot de passe pour ouvrir la base de données. Dès qu'un mot de passe est défini, une boîte de dialogue qui exige un mot de passe s'affiche lors de chaque ouverture de la base de données. Seuls les utilisateurs qui tapent le mot de passe correct pourront ouvrir la base de données.

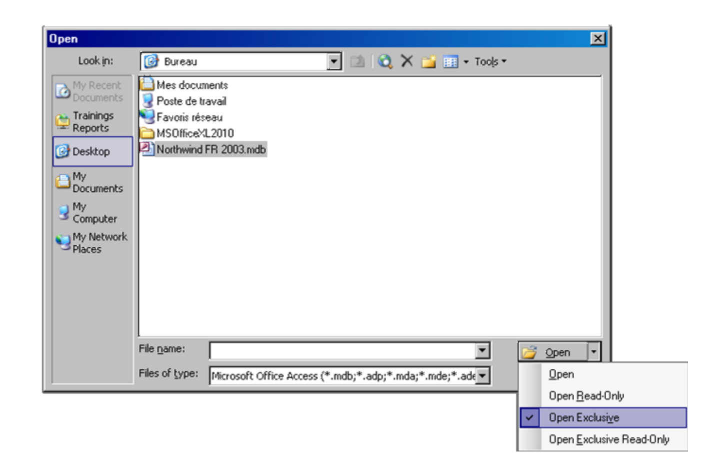
Cette méthode est relativement sûre (Microsoft Access code le mot de passe pour qu'il ne soit pas accessible en lisant directement le fichier de base de données), mais elle ne s'applique qu'à l'ouverture d'une base de données. Dès qu'une base de données est ouverte, tous ses objets sont à la disposition de l'utilisateur. Pour une base de données qui est partagée entre un petit groupe d'utilisateurs, ou sur un ordinateur isolé, définir un mot de passe est souvent suffisant.

Marche à suivre:

1. Fermer la base de données. Si la base de données est partagée sur un réseau, demander aux autres utilisateurs de fermer la base de données.

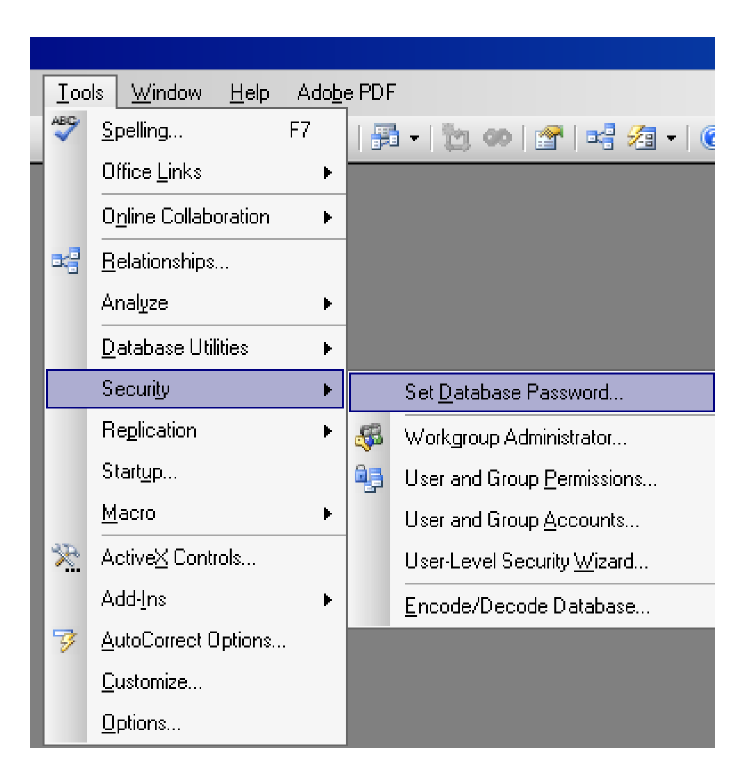
2. Faire une copie de sauvegarde de la base de données et la stocker dans un endroit où elle sera en sécurité !

3. Dans le menu *Fichier*, cliquer sur *Ouvrir*... Activer ensuite la case à cocher *Mode exclusif* dans le bouton *Ouvrir* en bas à droite de la boîte de dialogue. Ce qui aura pour effet d'ouvrir la base de données mais pas en mode exclusif…:

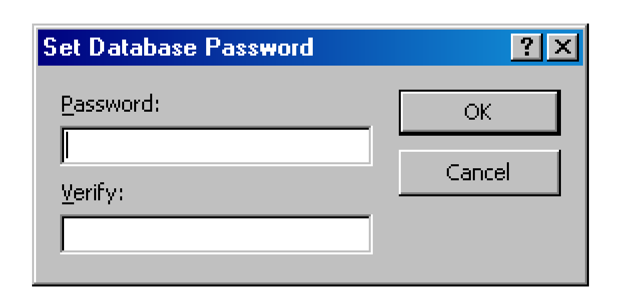


4. Refermez la base de données et fait un simple *Fichier/Ouvrir* pour rechoisir la même base de données.

5. Ensuite, allez dans le menu *Outils*/Sécurité/*Définir le mot de passe de base de données*:



6. Dans la zone *Mot de passe*, taper le mot de passe et le confirmer:



7. Valider.

Le mot de passe est à présent défini. La prochaine fois que tout utilisateur ouvrira la base de données, une boîte de dialogue demandant le mot de passe s'affichera.

Attention:

- Un mot de passe respecte la casse on doit le taper exactement comme il a été défini (maximum 20 caractères).
- Si on perd ou si on oublie le mot de passe, il ne peut pas être récupéré (sans logiciels spécialisés gratuitement disponibles sur le web) et plus personne ne pourra ouvrir la base de données !

Remarque: le danger avec cette méthode, c'est que chaque utilisateur a quand même accès à toute l'architecture de la base de données s'il connaît comment passer outre l'écran d'accueil (shift…). De plus, n'importe quel utilisateur pourrait l'ouvrir en "Mode exclusif" (*Fichier/Ouvrir* → Bouton *Ouvrir* → *Ouvrir en mode exclusif*) et dès lors empêcher chacun de ses collègues d'y écrire ou lire la moindre information !

17.2 Sécurité au niveau utilisateur

17.2.1 Protocole

Attention, si vous ne suivez pas le protocole suivant **SCRUPULEUSEMENT** en ce qui concerne une base sécurisée, vous allez rencontrer des complications!!!

1. Il faut créer les fichiers de sécurité *.mdw avant de créer sa base de données car le moteur JET l'utilise dès le début pour coder les droits (si malheureusement la base existe déjà, il faudra en créer une nouvelle au préalable en ayant respecté ce point et importer tous les objets dedans).
2. Fermer et rouvrir MS Access
3. Ensuite, dans *Outils/Sécurité/Gestion des utilisateurs et des groupes…*, définir tous les *Users* et *Groups* (minimum 1 User: MoiMeme, 1 Group: MonGroup
4. Faire toutes les associations *Users-Groups* (dont *MoiMeme-MonGroupe*)
5. Mettre un des nouveaux *Users* (MoiMeme) dans le groupe des *Administrateurs*
6. Enlever le User *Administrateur* du Groupe *Administrateurs*
7. Donner un mot de passe à l'*Administrateur*
8. Quitter Access
9. Ouvrir Access (pour mettre en place ces nouveaux paramètres)
10. S'annoncer avec le nom du nouveau user (MoiMeme) et aucun mot de passe en passant par *Outils/Sécurité/Comptes utilisateurs et groupes…*
11. Créer une base de données (ainsi ce sera MoiMeme qui sera le propriétaire de la base)

12. Faire *Outils/Sécurité/Autorisations d'accès* pour paramétrer les droits:

13. Retirer tous les droits du groupe des Utilisateurs (s'il est là)

14. Retirer tous les droits du groupe des Administrateurs

15. Mettre tous les droits au groupe des Développeurs (si vous créé un tel groupe…)

16. Et faire cela sur chacun des types d'objets de la base de données. Ainsi, la base de données ne pourra pas être ouverte pas l'utilisateur par défaut (utilisateur: Administrateur)

17. Construire sa base de données

18. Créer tous les utilisateurs, groupes avec droits ad hoc

19. Mettre la base à disposition des utilisateurs en tant que raccourci dans lequel il y aura une liaison avec le *.mdw (voir plus loin) se trouvant le réseau pour l'ouvrir de manière sécurisée quel que soit l'ordinateur

20. Enfin… faire un back up régulièrement du fichier *.mdw créé au point **1**

Si vous ne suivez pas ces étapes, n'importe qui pourra ouvrir votre base de données sans aucune sécurité sur un autre ordinateur.

17.2.2 Méthodes

La méthode la plus flexible et la plus étendue pour protéger une base de données s'appelle la "**sécurité au niveau utilisateur**". Ce type de sécurité est similaire aux méthodes utilisées dans la plupart des systèmes de réseau. Les utilisateurs doivent s'identifier et taper un mot de passe lorsqu'ils démarrent MS Access (et non pas nécessairement une base de donnée spécifique !!!).

Au sein du fichier d'informations de groupe de travail, ils sont identifiés comme étant les membres d'un groupe. MS Access fournit deux groupes par défaut: les administrateurs (appelés le "**groupe Administrateurs**") et les utilisateurs (appelés le "**groupe Utilisateurs**"), mais des groupes supplémentaires peuvent être définis (heureusement…).

Les autorisations d'accès sont accordées aux groupes et aux utilisateurs pour déterminer de quelle manière ils sont autorisés à travailler avec chaque objet dans une base de données. Par exemple, les membres du groupe utilisateurs pourraient être autorisés à visualiser, introduire ou modifier des données dans une table, mais ils ne pourraient pas changer la présentation de cette table. Le *groupe Utilisateurs* pourrait être autorisé à visionner les données de certaines tables et se voir refuser l'accès total à une table contenant des données sensibles. Les membres du *groupe Administrateurs* ont toutes les autorisations d'accès sur tous les objets d'une base de données.

Les trois raisons principales d'utilisation de la sécurité au niveau utilisateur sont:

- Protéger la propriété intellectuelle du code.

- Eviter que les utilisateurs ne détériorent par inadvertance une application en changeant le code ou les objets dont l'application dépend.
- Protéger des données essentielles dans la base de données.

Les groupes, les utilisateurs, les bases de données et toutes les autorisations d'accès font partie d'un **environnement commun** qu'on appelle un **espace de travail** (Workspace).

La sécurité au niveau utilisateur s'applique donc au niveau du logiciel MS Access et non uniquement sur une unique base prédéfinie !!!

Remarque: Bien qu'un utilisateur appartienne toujours à un groupe, s'il a des droits différents du groupe, ce seront ses droits individuels actifs qui prendront le dessus sur ceux inactifs du groupe (et si le groupe a donc des éléments actifs qui sont inactifs chez l'utilisateur, alors il héritera automatiquement des droits actifs du groupe).

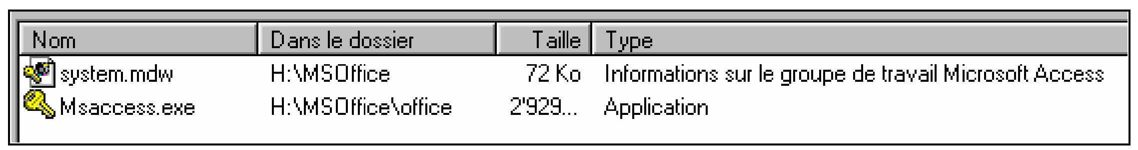
Dans MS Access, le système de sécurité est **toujours en activité**:

- on appartient toujours à un espace de travail !
- on fait toujours partie d'un groupe !
- on est toujours un utilisateur !

Ces diverses informations sont stockées dans un fichier discret, mais très important, que l'on trouve au premier niveau de l'environnement d'installation de MS Access, soit dans MSOFFICE (**ou directement dans le moteur JET de votre base si le protocole précédemment présenté a été suivi**) pour les machines en réseau, soit dans:

C:\Documents and Settings\<Utilisateur>\Application Data\Microsoft\Access

pour les machines autonomes: le fichier **system.mdw.**



Nom	Dans le dossier	Taille	Type
system.mdw	H:\MSOffice	72 Ko	Informations sur le groupe de travail Microsoft Access
Msaccess.exe	H:\MSOffice\office	2'929...	Application

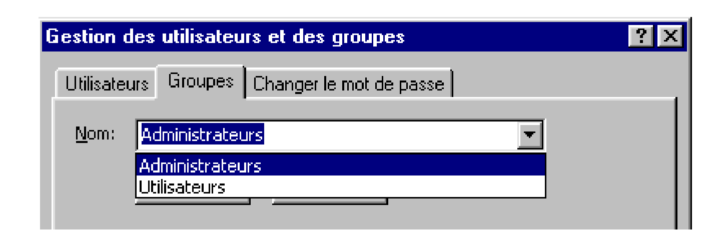
MDW = Microsoft Data Workspace

C'est le "fichier d'informations du groupe de travail" comme l'appelle Microsoft. Ce fichier est créé au moment de l'installation du logiciel sur la machine. Les paramètres nécessaires à la génération de ce fichier sont pris dans les deux informations que l'on doit fournir lors de l'installation: le nom de l'utilisateur de la machine et le nom de l'organisation dans laquelle il travaille.

Par défaut, toute personne qui lance MS Access va se trouver englobée dans **l'espace de travail** défini par ce fichier *System.mdw:*

1. Cet espace contient deux groupes par défaut:

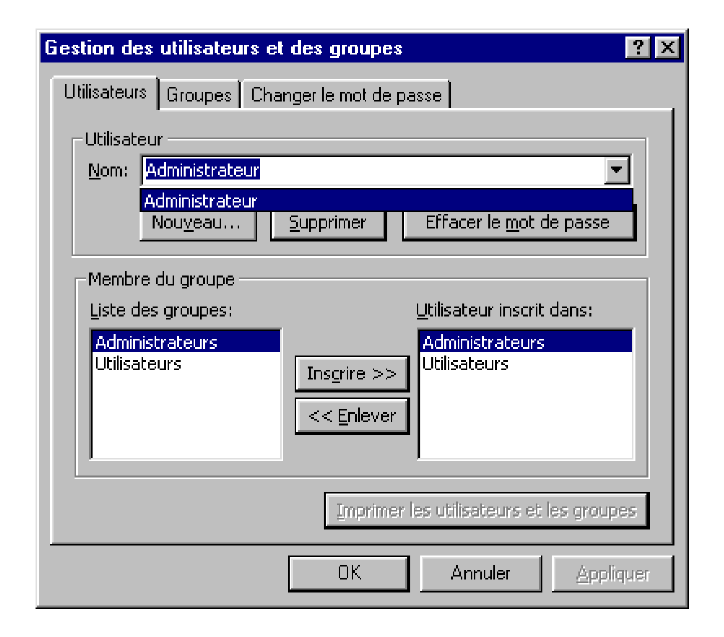
   1. les **Administrateurs** (Admins)
   2. les **Utilisateurs** (Users)



2. Cet espace abrite deux utilisateurs:

1. Un **Administrateur** (Admin)

2. Un **Utilisateur** standard (Guest)



Remarques:

- Dans la version française, les termes anglais ont été traduits, mais il faut utiliser les termes anglais en Visual Basic Application (V.B.A.).

- Il arrive que l'utilisateur standard (Guest) ne soit pas créé dans l'espace de travail. On n'aura donc que l'Administrateur (Admin) à disposition…

Le(s) utilisateur(s) par défaut susmentionné(s) n'ont pas de mot de passe comme nous l'avons déjà implicitement mentionné dans le protocole de début !

17.2.3 Modélisation des droits

Pour définir les groupes d'utilisateurs il faut d'abord définir une stratégie d'utilisation de l'application. Le mieux est d'organiser tout cela sous forme de tableau.

Voici l'exemple d'un tableau réalisé. Notez que n'importe quel tableur ou logiciel permettant de faire des tableaux fait l'affaire.

L: Lecture
A: Ajout
M: Modification
S: Supression

Tableau simple de résumé de la sécurité

	Objets	Table Clients				Librairie Rapports				Site Projet				
	Droits	L	A	M	S	L	A	M	S	L	A	M	S	
Groupes	Contributeurs		o								o			Membres:
	Designers						o		o					Membres:
	Visiteurs		o										o	Membres:
	Commerciaux			o				o			o			Membres:
	Acheteurs										o			Membres:
	Finance		o		o		o							Membres:
Utilisateurs (hors groupes)	Bill Gates		o					o						
	Vincent Isoz			o		o				o		o		

17.2.4 Dangers de l'espace de travail "par défaut":

Principe: Quand on lance MS Access sans indication particulière:

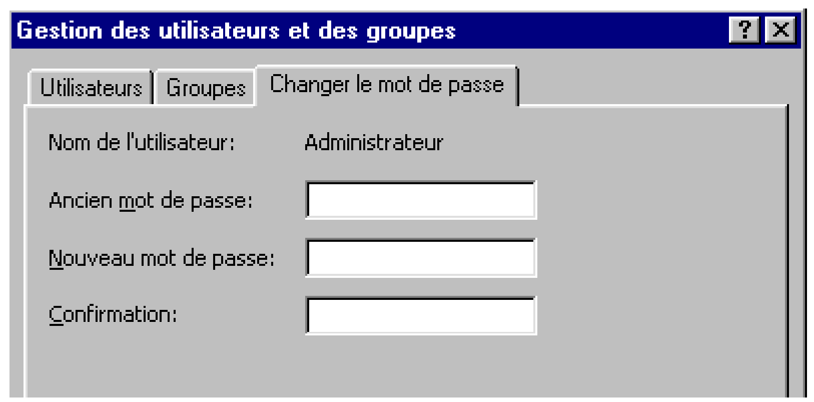
- on utilise toujours l'espace de travail par défaut décrit pas System.mdw
- on est toujours considéré comme l'Administrateur de cet espace de travail !!

Conséquence:

En tant qu'Administrateur, on a tous les pouvoirs sur l'espace de travail en cours, ainsi que sur toutes les bases de données créées **à partir de cet espace** et, bien entendu, sur tous les objets que peuvent contenir ces bases !

Risque (qui concernant les O.S. avant Windows XP à ma connaissance…):

Le premier utilisateur un peu futé et mal intentionné (ou maladroit…) attribue un mot de passe à l'Administrateur … il devient le propriétaire de l'espace de travail et, à partir de cet instant, il empêche tous les autres utilisateurs … d'utiliser toutes les bases de données déjà créées à partir de cet espace de travail (de son ordinateur au fait)!



Changer le mot de passe ne permet de le faire que pour l'utilisateur connecté

Précautions à prendre:

1. **Suivre le protocole donné en début de chapitre!**

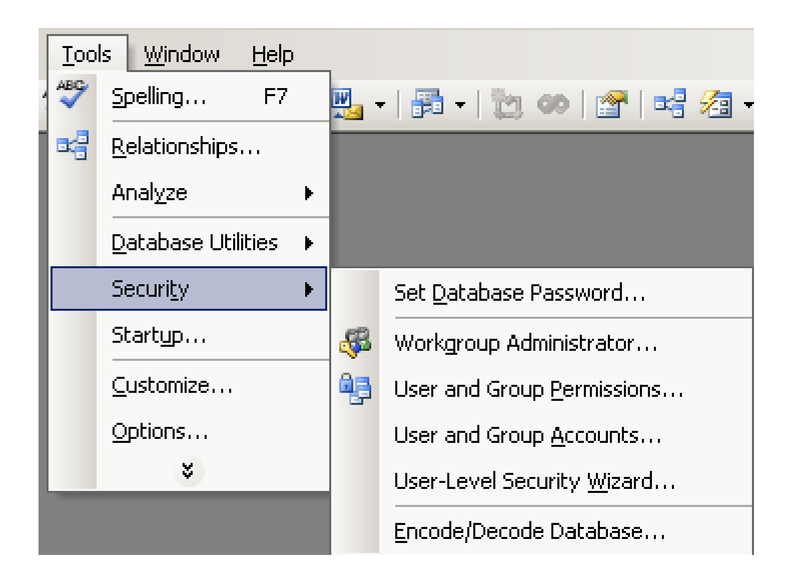
2. Interdire la modification du fichier System.mdw aux utilisateurs simples (dossier sur disque réseau avec droits définis via Active Directory).

3. Garder une copie du fichier System.mdw en lieu sûr pour remplacer la version qui serait modifiée par un utilisateur malveillant (donc in extenso toute personne qui sait où se trouve le fichier mdw d'une base pour le réinitialiser en l'écrasant par System.mdw après avoir changé le nom du fichier…. d'où le fait que beaucoup cherchent à mettre ce fichier sur un disque réseau en lecture seule)

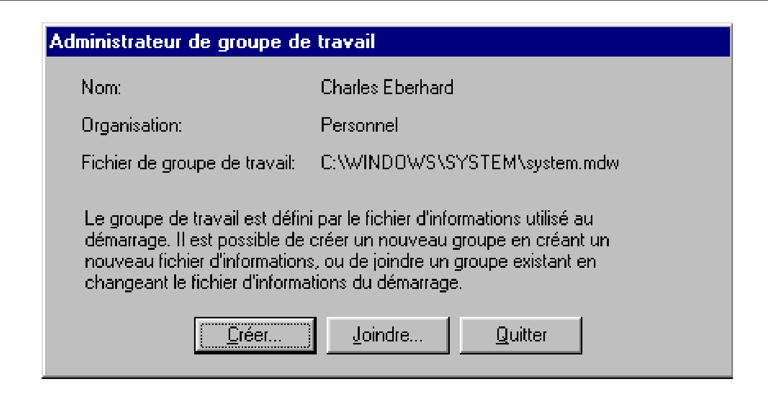
4. Organiser et administrer un ou plusieurs espaces de travail bien protégés.

Création d'un nouvel espace de travail:

Il faut exécuter le programme WRKGADM.EXE (**WoRK Group ADMinistrator**) qui se trouve, soit dans Windows\System, ou en passant par l'environnement MS Access par *Tools/Security/Workgroup Administrator*:



pour pouvoir créer un nouveau fichier System.mdw.

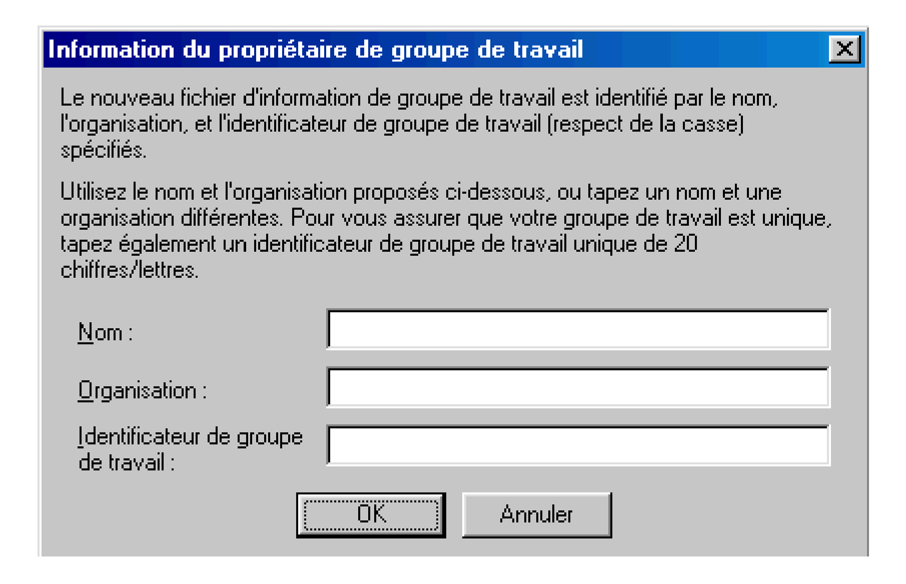


Comme on le voit, ce fichier n'est pas tiré du néant, mais est "copié" à partir d'un fichier existant: cela peut être l'original ou déjà une version préparée par nos soins conformément au protocole susmentionné.

Remarque: La dénomination "groupe de travail" n'est pas très heureuse: il vaudrait mieux parler d'un "espace de travail", terme utilisé dans Visual Basic. Le mot groupe peut prêter à confusion, car on va définir des groupes d'utilisateurs à l'intérieur du "groupe de travail"…

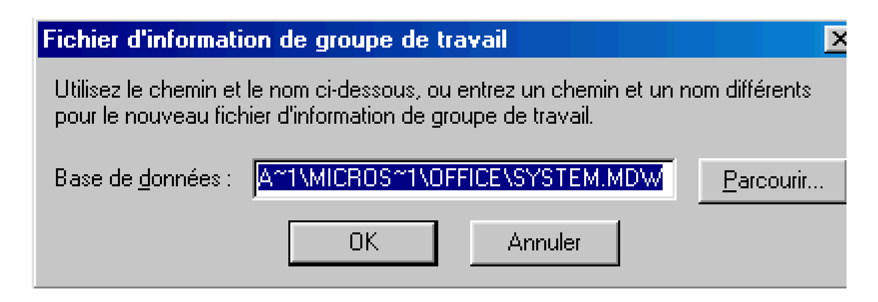
Trois informations sont nécessaires pour générer le nouveau fichier:

1. Le nom
2. L'organisation
3. L'identificateur de groupe de travail.

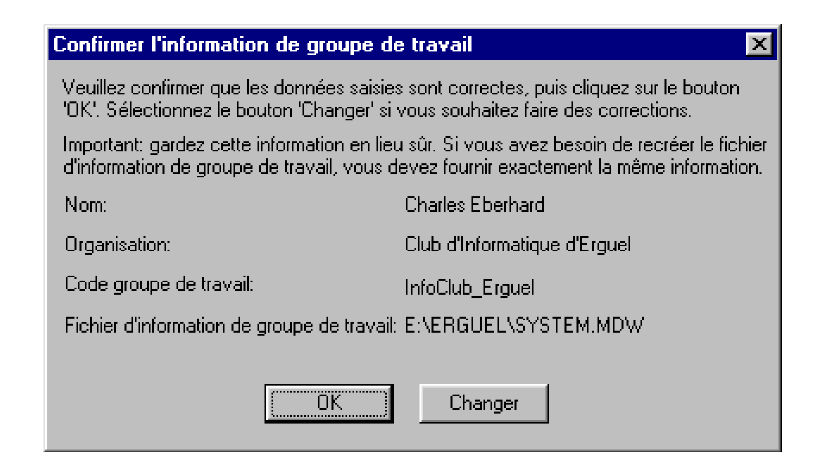


L'*Identificateur de groupe de travail* doit assurer que l'espace de travail que l'on crée est réellement **unique**, car la même personne (Nom), dans la même entreprise (Organisation), peut créer plusieurs "groupes de travail". Ce code est donc le seul moyen de les distinguer !

Ensuite, le logiciel va nous demander où le sauvegarder et sous quel nom:



On peut donner un nouveau nom au fichier créé et le stocker dans le même répertoire que le fichier *System.mdw* d'origine, ou enregistrer le nouveau *System.mdw* dans un répertoire particulier qui sera partagé par les différents utilisateurs de ce "groupe de travail" (cas le plus communément utilisé).



Il est vital de conserver les paramètres utilisés, par écrit et dans un endroit sûr.

En effet, si le fichier "d'informations du groupe de travail" venait à être détruit ou corrompu, la seule façon de pouvoir accéder aux bases de données partageant cet espace de travail - si la sécurité est correctement mise en place (avec l'assistant de sécurité par exemple) - est de reconstruire un fichier identique !

Attention: Il faut dès lors absolument respecter les majuscules et les minuscules d'origine.

Remarque: Si le fichier *.mdw est créé de manière classique (c'est-à-dire sans l'assistant), une simple suppression de celui-ci permettra à tout le monde d'accéder à la base de données avec tous les droits.

Nous avons maintenant plusieurs espaces (groupes) de travail. Comment faire pour indiquer celui avec lequel on veut travailler ? Dans sa version française, Microsoft parle de "rejoindre un groupe de travail Microsoft Access".

Deux méthodes sont possibles:

<u>Première méthode:</u> Utiliser le programme "Administrateur de groupe de travail"

1. Démarrer Wrkgadm.exe (l'Administrateur de groupe de travail) qui doit se trouver dans le sous-dossier Système du dossier Windows de la machine intéressée par les contraintes de sécurité ou passer par le menu *Outils/Sécurité/Administrateur de groupe de travail*.

2. Dans la boîte de dialogue, cliquer sur *Joindre*.

3. Taper le chemin d'accès et le nom du fichier d'informations de groupe de travail qui définit le groupe de travail MS Access que l'on veut rejoindre ou utiliser "Parcourir" pour localiser et sélectionner le fichier d'informations du groupe de travail désiré.

Ce chemin a été sauvegardé dans la base de registres sous la clé:

Hkey_Local_Machine\Software\Microsoft\Office\<Version actuelle d'Office>\Access\Jet\<Version actuelle du Jet>\Engines



Au démarrage suivant, Microsoft Access utilisera automatiquement les informations stockées dans le fichier .MDW du groupe de travail que l'on vient de rejoindre.

<u>Deuxième méthode:</u> Démarrer MS Access avec une option sur la ligne de commande (niveau expert en connaissances de MS Windows).

Il suffit d'ajouter l'option /wrkgrp à la ligne de commande.

Exemple: F:\MsOffice\Office\Msaccess.exe /wrkgrp E:\Erguel\Erguel.mdw

On peut taper cette ligne dans la fenêtre "CMD".

<u>Attention:</u> Avec des machines en réseau qui sont utilisées par des personnes différentes, la deuxième méthode est nettement préférable. En effet, la base de registres **Hkey_Local_Machine n'est pas enregistrée avec les paramètres de l'utilisateur, si bien que l'information va demeurer dans la machine quel que soit l'utilisateur!!!** Le prochain utilisateur qui va lancer Access sera d'emblée connecté au dernier groupe de travail utilisé et non pas au groupe de travail par défaut !

Remarque: Dans MS Access 97 et ultérieur, les préférences utilisateur sont stockées dans la base de registres Windows sous la clé:

Hkey_Current_User\Software\Microsoft\Office\<Version actuelle d'Office>\Access\Settings.

Maintenant que l'espace de travail est créé, il faut s'occuper de l'organiser:

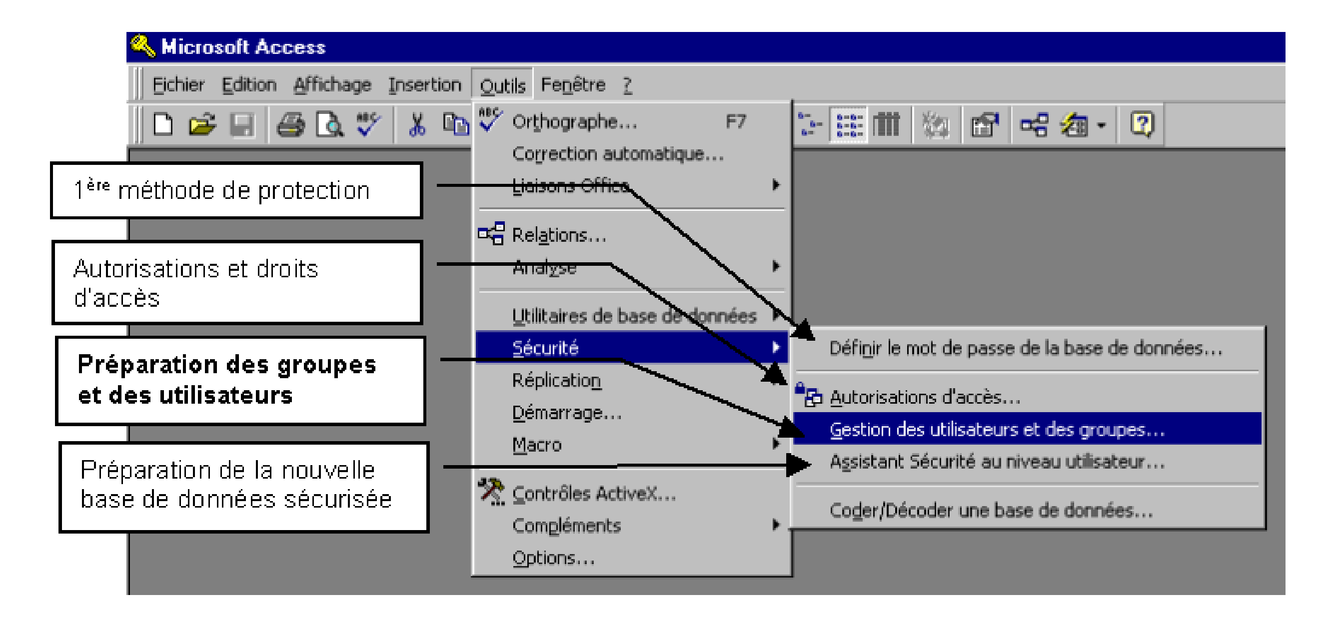
1. Définir les différents groupes d'utilisateurs

2. Identifier les différents "comptes" utilisateurs

3. Attribuer les utilisateurs à un ou plusieurs groupes.

Ces différentes opérations s'effectuent dans MS Access, une fois que l'on est "connecté" au groupe de travail désiré !

C'est le Menu *Outils* qui donne accès aux options *Sécurité*.



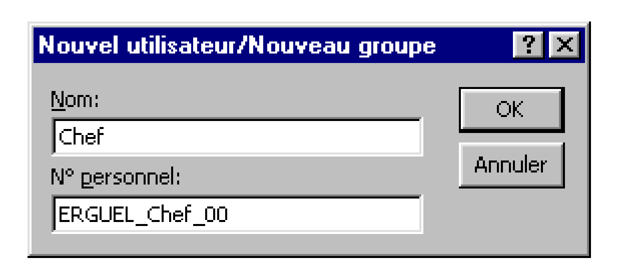
Rappel: Lorsque l'on crée pour la première fois un fichier d'information de groupe de travail, on est considéré par le système comme étant le fameux "Administrateur" qui dispose de tous les droits. Il faut user de ces droits avec discernement et agir avec méthode pour ne pas se trouver coincé par une mauvaise manipulation.

1ère opération (déjà mentionné au début de ce chapitre):

Créer un nouvel Administrateur, différent de l'administrateur par défaut (piège classique)!

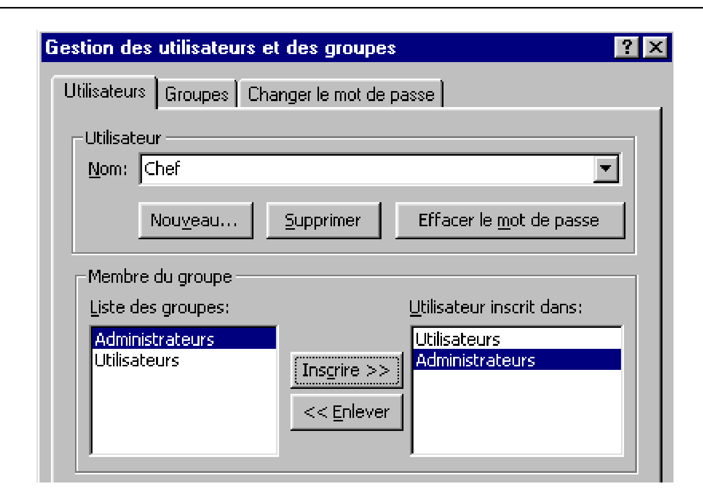
Choisir un nom symbolique du genre: BigBoss, Patron, Chef, etc. ou utiliser le nom du véritable administrateur du groupe de travail

Le N° personnel permet de distinguer des utilisateurs qui auraient le même nom. Comme on l'a déjà vu, Il faut conserver cette information dans un endroit sûr, pour le cas où il faudrait recréer le fichier d'information du groupe de travail ainsi que les utilisateurs qui le composent !



2ème opération:

Inscrire absolument cet utilisateur dans le *groupe des Administrateurs* !



En prévision de la 3ème opération, il est indispensable bien évidemment qu'une personne au moins soit membre du groupe "Administrateurs" pour disposer des pouvoirs nécessaires à la poursuite des opérations !

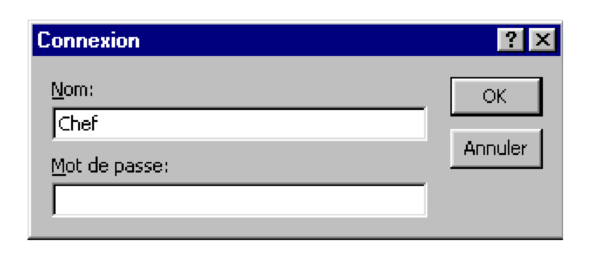
3ème opération:

Il faut maintenant retirer l'ancien Administrateur du groupe "Administrateurs" (sinon vous faites un travail pour rien). De cette manière, les éventuels "Administrateurs" d'autres groupes de travail ne pourront plus jamais prendre possession des bases de données qui vont être créées par le nouveau groupe de travail.

Il faut pour cela donner un mot de passe à cet ancien "Administrateur", car il n'est pas possible de détruire ce compte (? !) qui va demeurer maintenant en tant qu'utilisateur. Il faut donc empêcher que n'importe qui puisse accéder, sans contrôle, à ce groupe de travail en utilisant le nom "Administrateur".

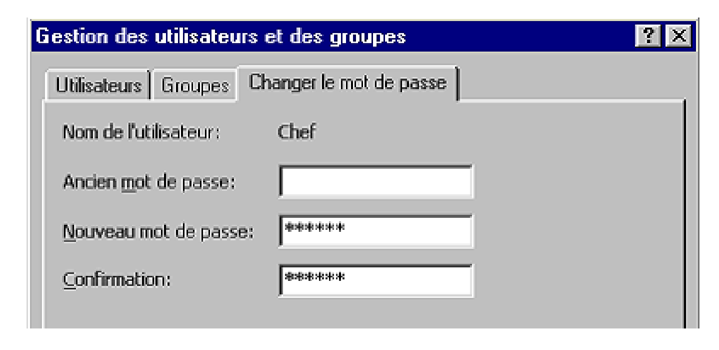
4ème opération:

Quitter et relancer MS Access. Cette fois, une boîte de dialogue "Connexion" doit apparaître à l'écran… Il faut saisir ou sélectionner le nouvel "Administrateur". Il n'a pas encore de mot de passe …



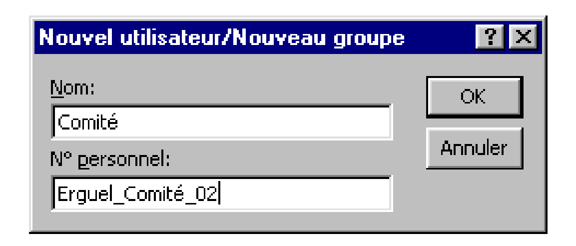
5ème opération:

Attribuer un mot de passe à l'Administrateur, puisque l'on est connecté sous son nom.



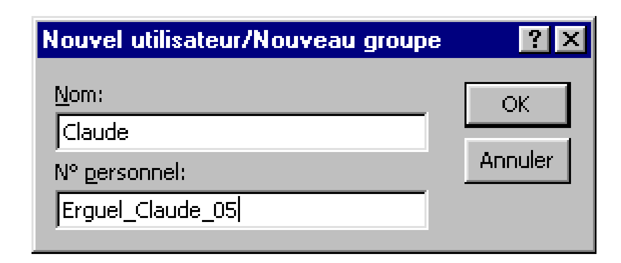
6ème opération: créer les groupes d'utilisateurs.

Par exemple: Membres, Comité, etc. selon la structure et les besoins de l'entreprise…



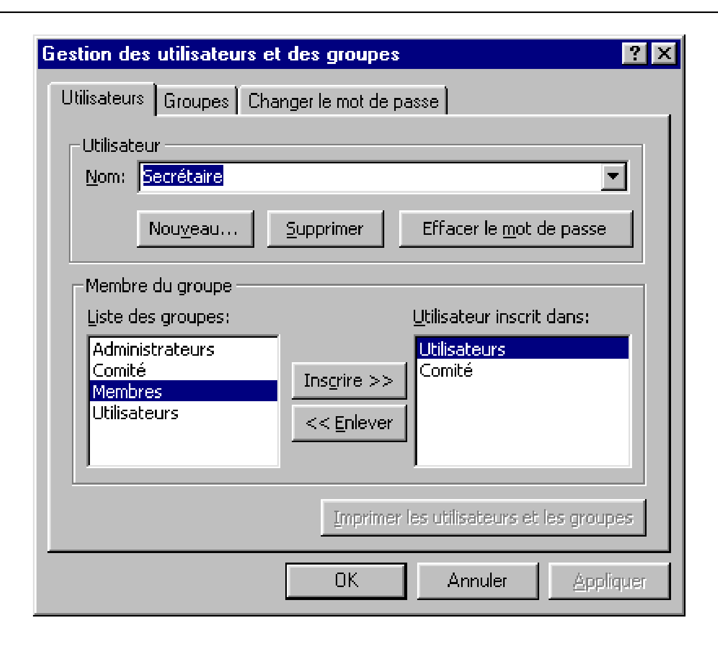
7ème opération: créer les comptes d'utilisateurs.

Le terme "compte d'utilisateur" provient du début de l'informatique, lorsque l'on tenait le compte du temps que chaque utilisateur passait sur son terminal connecté sur l'ordinateur central en "time sharing".



8ème opération: attribuer les utilisateurs aux différents groupes.

L'avantage des groupes, c'est qu'en attribuant un utilisateur à l'un d'entre eux, cet utilisateur acquière d'un seul coup tous les droits et toutes les autorisations que l'on a accordés à ce groupe au préalable.



Il n'y a donc pas besoin de définir individuellement le détail des droits de chaque utilisateur: il suffit de le prévoir pour son groupe et de lui faire rejoindre ce groupe !

Remarque: Droits et autorisations seront traitées plus loin.

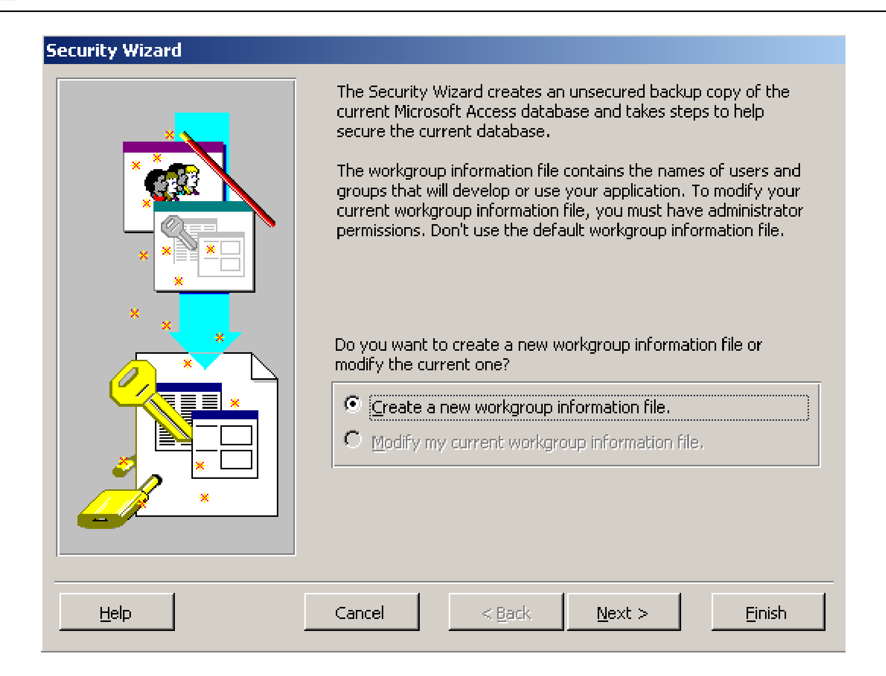
Maintenant que l'espace de travail est délimité, il faut encore y incorporer les bases de données que les utilisateurs vont utiliser ! Deux cas de figure non exclusifs peuvent se présenter:

- On veut créer de nouvelles bases de données à partir de l'espace de travail choisi.
- On veut utiliser, en les sécurisant dans cet espace de travail, des bases de données existantes.

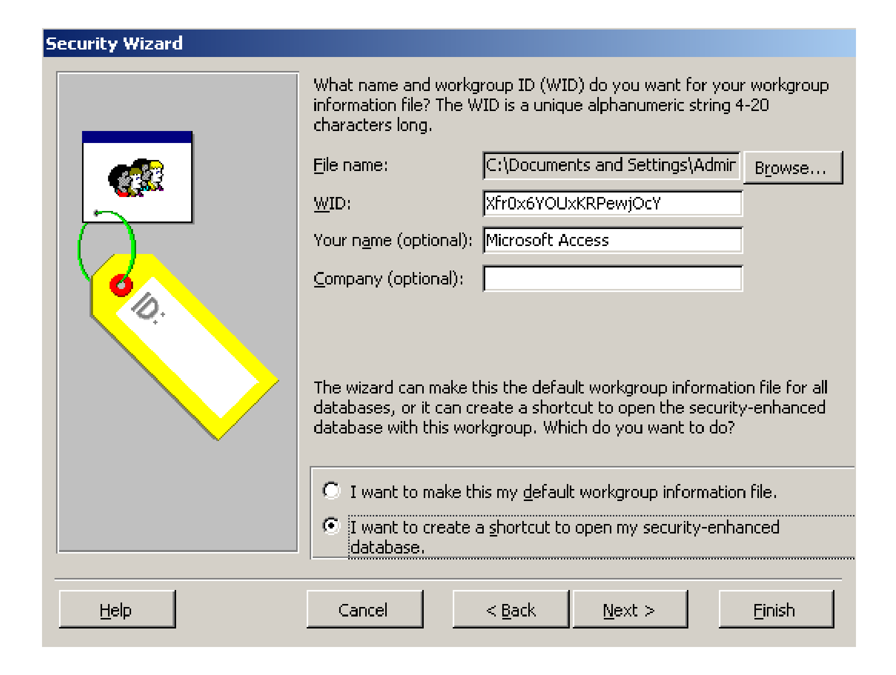
Dans les deux cas pourtant la procédure est la même: il faut faire une copie de la base de données existante en la passant par le protocole vu plus haut ou par une moulinette particulière: un "Assistant sécurité" qui a été apparu avec Access 2002.

Donc pour sécuriser une base de données existante une autre possibilité plus simple que le protocole consiste à utiliser ce fameux assistant (qui cependant n'efface pas les groupes et utilisateurs dont tout le monde connaît les noms à l'avance… ce qui n'est pas vraiment très très astucieux en termes de sécurité…):

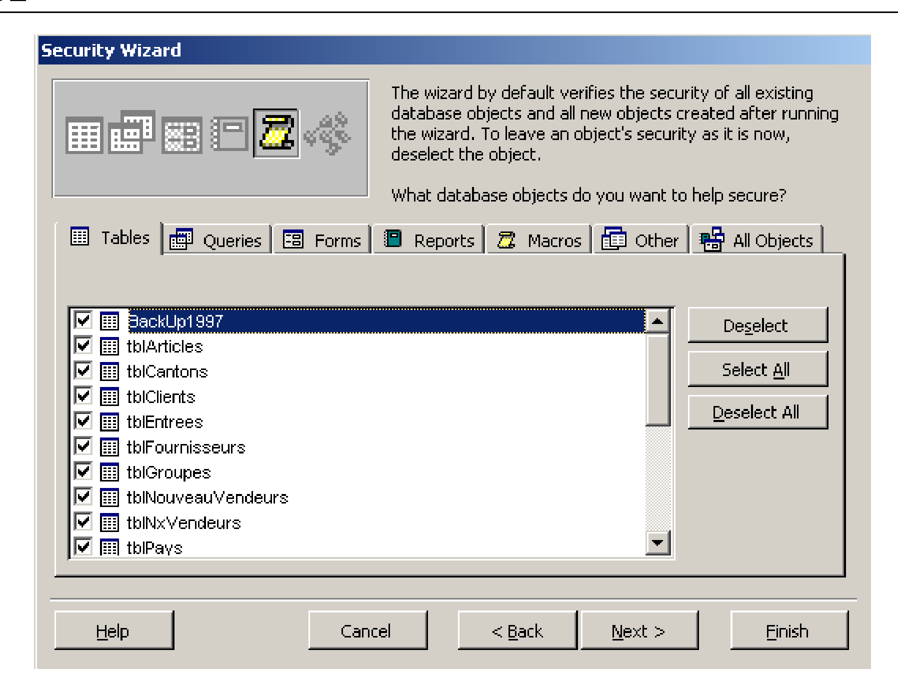
1. Se connecter à l'espace de travail en tant qu'*Administrateur* .
2. Ouvrir la base de données (pleine ou vide).
3. Menu *Outils/Sécurité/Assistant sécurité au niveau utilisateur*



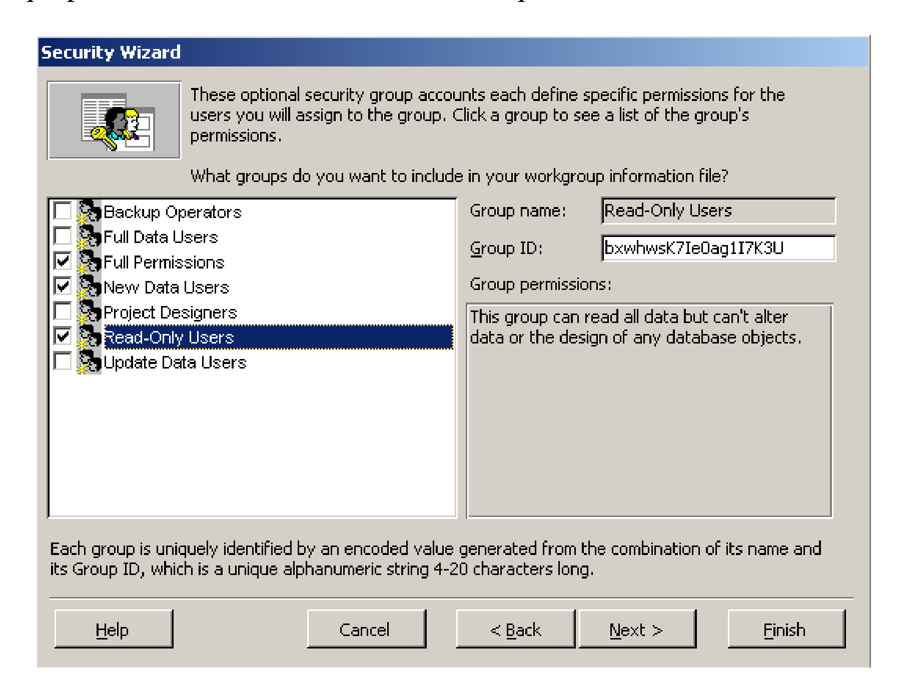
On clique sur *Next*:



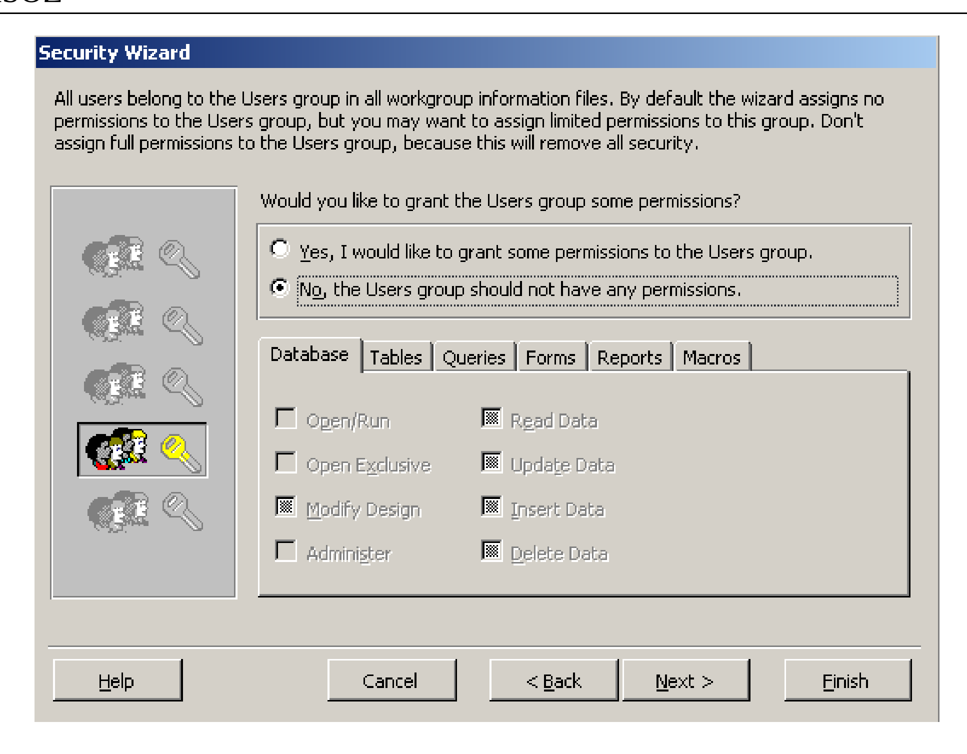
On clique sur *Next*:



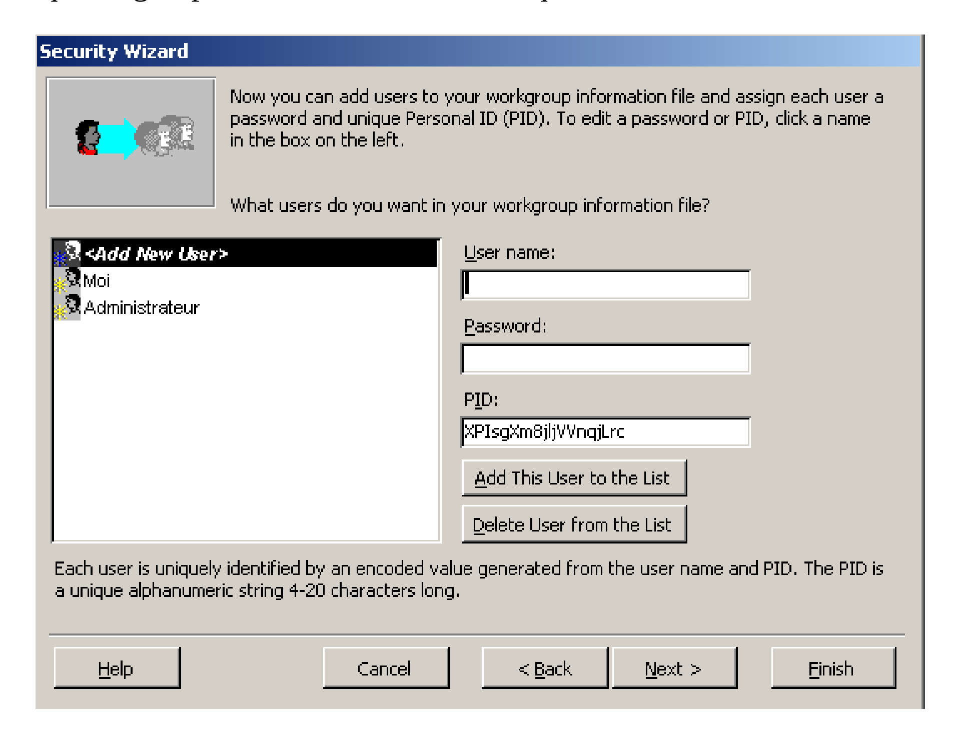
on choisit ce qui pourra être sécurisé ou non. On clique sur *Next*:



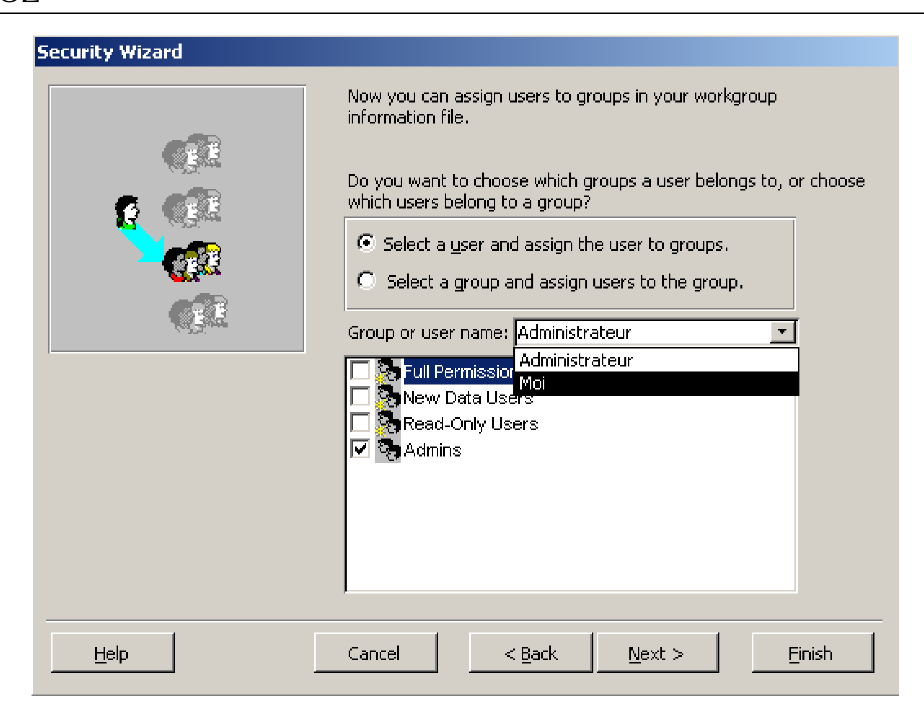
On choisir les groupes types d'utilisateurs que l'on voudra pouvoir utilisateur par la suite (voir la description plus bas). Et on clique sur *Next*:



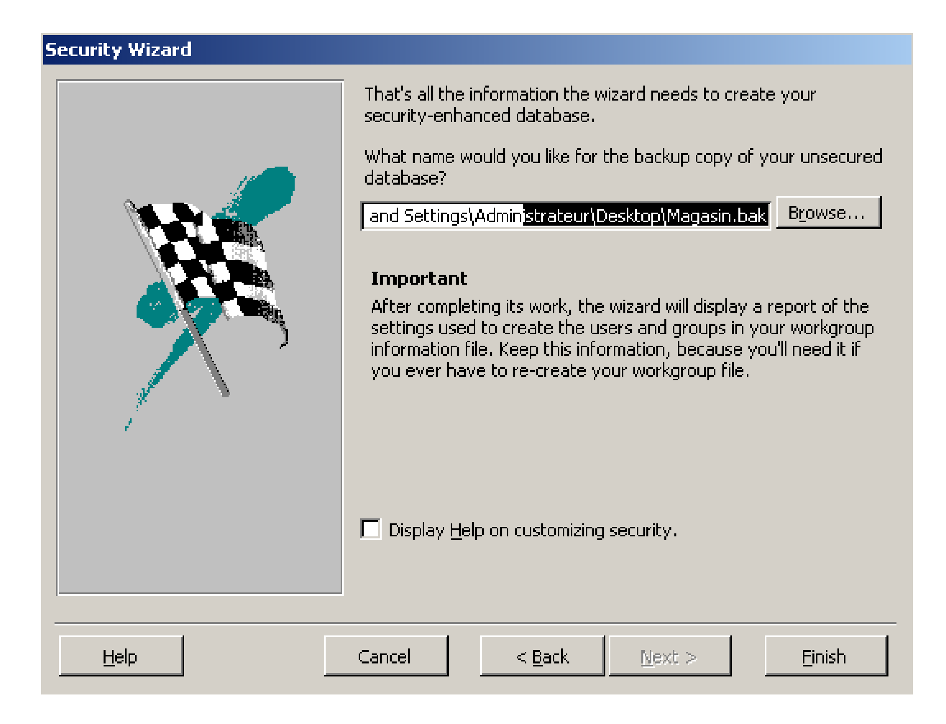
Il nous demande si on veut faire quelque chose de particulier avec le *Groupe Utilisateurs* qui a un statut un peu spécial dans Access puisqu'il existe sur tous les PCs. Il vaut mieux ne rien changer afin que ce groupe n'ait aucun droit. On clique sur *Next*:



On créée des utilisateurs et on clique sur *Next*:



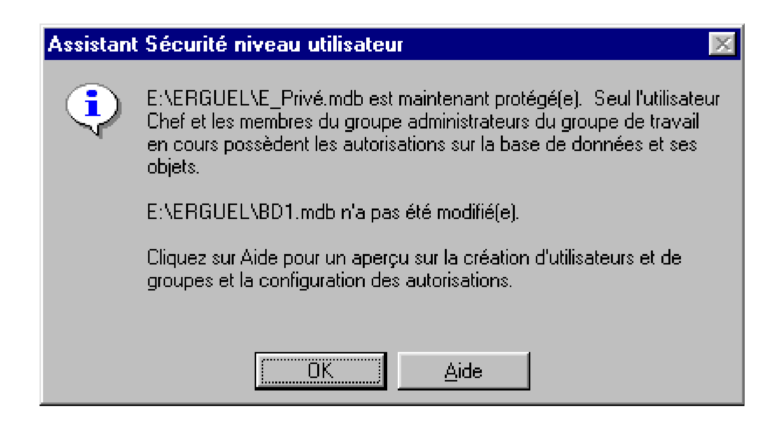
On assigne chaque utilisateur à des groupes spécifiques de sécurité. On clique sur *Next*:



Il propose de faire une copie non sécurisée de la base de données (heureusement car certains oublient…). On clique sur *Finish* et on aura dans notre dossier de travail, la base Access avec son espace de travail (*.mdw) + un raccourci spécial pour l'ouvrir:



La base ainsi créée est la propriété de l'*Administrateur*: tant qu'il n'aura pas déterminé les droits d'accès, nul autre ne sera autorisé à pénétrer et, à plus forte raison, à utiliser cette base de données:



Donc l'assistant redéfinit directement la sécurité dans la structure JET de la base de données !

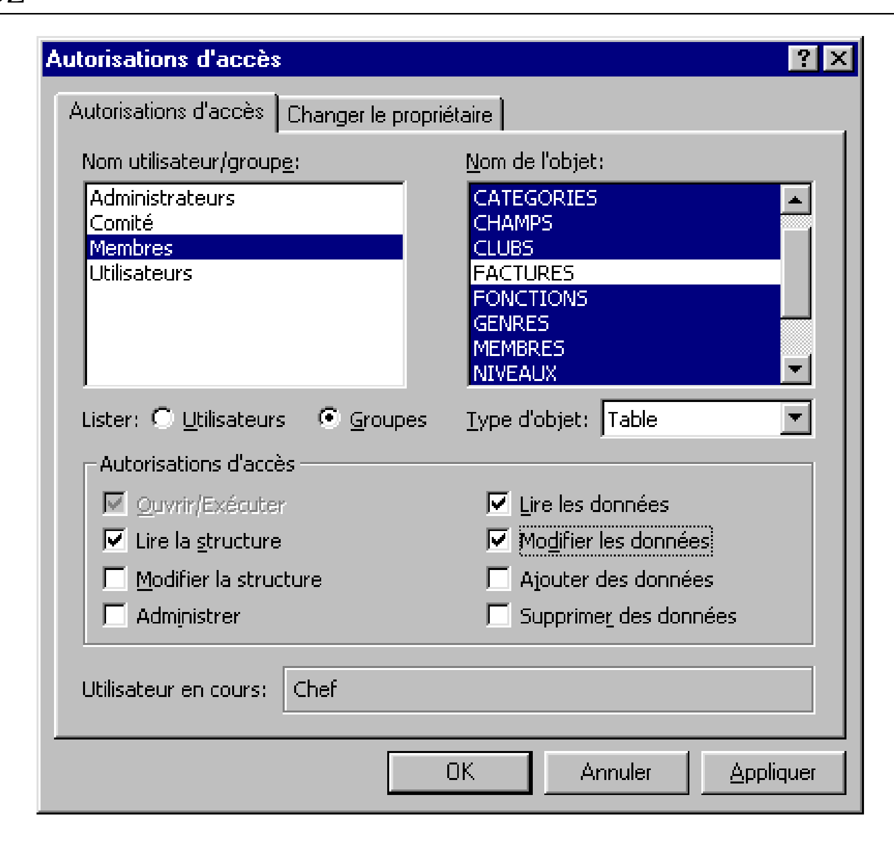
Si l'on veut gérer correctement une base de données sécurisée, il faut déterminer pour chaque utilisateur et pour chaque groupe quels sont leurs droits précis sur chaque objet de la base!

La meilleure méthode est de définir les droits et les autorisations pour un groupe, d'assigner un utilisateur à ce groupe et de tester si les manipulations des données nécessaires sont toutes possibles pour cet utilisateur. En cas de problèmes, modifier les autorisations du groupe. Quand tout fonctionne, tous les autres utilisateurs assignés à ce groupe pourront travailler sans soucis.

1. Menu *Outils/Sécurité/Autorisation* d'accès
2. Dans l'onglet *Autorisations* d'accès, choisir *Groupes* (ou Utilisateurs), puis le groupe (ou l'utilisateur) auquel on souhaite accorder les autorisations d'accès.
3. Cliquer sur le type d'objet dans la boîte *Type de l'objet*, puis sur le nom de l'objet auquel les autorisations d'accès se rapporteront dans la boîte *Nom de l'objet*.

Conseil: On peut sélectionner plusieurs objets dans la boîte *Nom de l'objet* en faisant glisser le pointeur de la souris sur les objets que l'on veut sélectionner ou en maintenant la touche CTRL enfoncée et en cliquant sur les objets souhaités.

Dans *Autorisations d'accès*, sélectionner les autorisations d'accès que l'on veut accorder, ou effacer les autorisations d'accès que l'on veut retirer au groupe ou à l'utilisateur, puis cliquer sur *Appliquer*.



Répéter les étapes 4 et 5 pour accorder ou retirer au groupe (ou à l'utilisateur) en cours d'autres autorisations d'accès portant sur d'autres objets.

Remarques:

- Certaines autorisations d'accès s'accompagnent automatiquement de la sélection d'autres autorisations d'accès. Par exemple, l'autorisation d'accès *Modifier les données* d'une table s'accompagne automatiquement des autorisations d'accès *Lire les données* et *Lire la structure*, car on en a besoin pour modifier les données d'une table (logique…)

- *Lire les données* s'accompagne automatiquement de *Lire la structure*. Dans le cas des macros, *Lire la structure* s'accompagne automatiquement de *Ouvrir/Exécuter*.

- Lorsque l'on modifie un objet et qu'on l'enregistre, il conserve les autorisations d'accès qui lui sont accordées (heureusement…)

- Toutefois, si un objet est enregistré sous un nouveau nom à l'aide de la commande *Enregistrer* sous dans le menu *Fichier*, ou en coupant et en collant, en important ou en exportant l'objet, les autorisations d'accès associées sont perdues ! Il faut les accorder à nouveau.

Types d'autorisations d'accès:

Autorisation	Permet à un utilisateur de	S'applique à
Ouvrir/exécuter	Ouvrir une base de données, un formulaire ou un état, ou	Bases de données, formulaires, états et macros

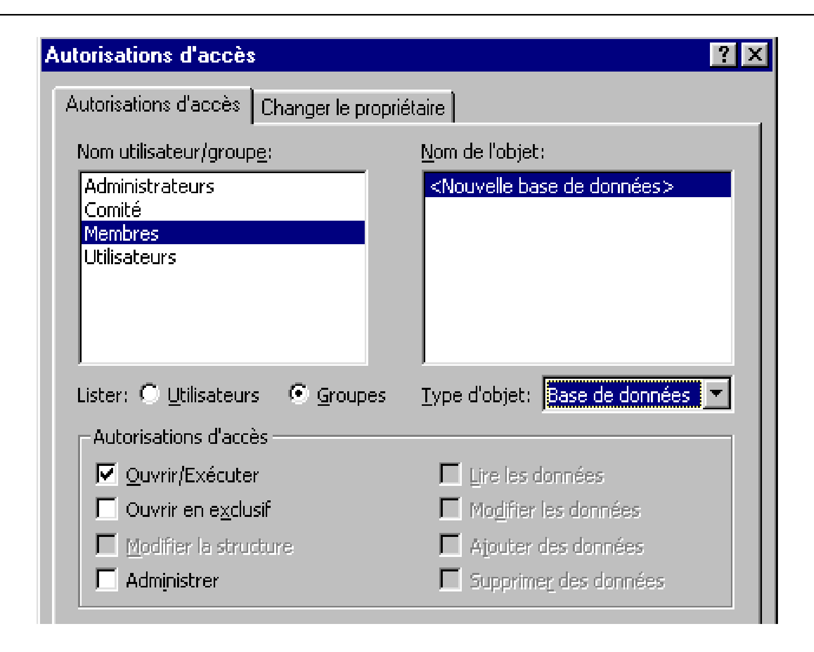
Autorisation	Permet à un utilisateur de	S'applique à
	exécuter une macro.	
Ouvrir en mode exclusif	Ouvrir une base de données avec accès exclusif.	Bases de données
Lire la structure	Afficher des objets en mode Création.	Tables, requêtes, formulaires, états, macros et modules
Modifier la structure	Afficher et modifier les objets, ou les effacer.	Tables, requêtes, formulaires, états, macros et modules
Lire les données	Afficher des données.	Tables et requêtes
Modifier les données	Afficher et modifier des données, sans pouvoir en ajouter ou en supprimer.	Tables et requêtes
Ajouter des données	Afficher et ajouter des données, sans pouvoir en modifier ou en supprimer.	Tables et requêtes
Supprimer des données	Afficher et supprimer des données, sans pouvoir en modifier ou en ajouter.	Tables et requêtes
Administrer	Définir un mot de passe pour une base de données, copier une base de données et modifier les propriétés de démarrage	

Tableau 7 Autorisations d'accès à une base

Ne pas oublier …

- D'enlever systématiquement toute autorisation de chaque objet de la base de données (tables, requêtes, formulaires, macros, etc.) pour le groupe *Utilisateurs* si on ne l'utilise pas en tant qu'Administrateur et seul utilisateur !

- D'accorder l'autorisation "Ouvrir/Exécuter" à l'objet "Nouvelle base de données" à chaque groupe, de telle manière que ses membres puissent ouvrir la base dans laquelle ils vont travailler !



Une fois ceci fait, reste l'aspect utilisateur. La connexion réclame plusieurs conditions:

1. Faire partie de l'espace de travail
2. Fournir son nom d'utilisateur
3. Eventuellement, fournir le nom de la base de données que l'on veut utiliser.

Il est possible d'automatiser cette procédure pour un utilisateur particulier au moyen d'un raccourci sur le bureau ou dans la barre des tâches.

Exemple 1: Choix de l'espace de travail et indication du nom de l'utilisateur

F:\MsOffice\Office\Msaccess.exe /wrkgrp G:\Erguel\Erguel.mdw /user Chef

Programme Access Groupe de travail Utilisateur

Le raccourci n'est pas obligé de contenir la partie avec */user Chef* !

Exemple 2: Idem + indication de la base de données

F:\MsOffice\Office\Msaccess.exe G:\Ecole\Ecole.mdb /wrkgrp G:\Ecole\System.mdw /user Bigboss

Programme Access Base Groupe de travail Utilisateur

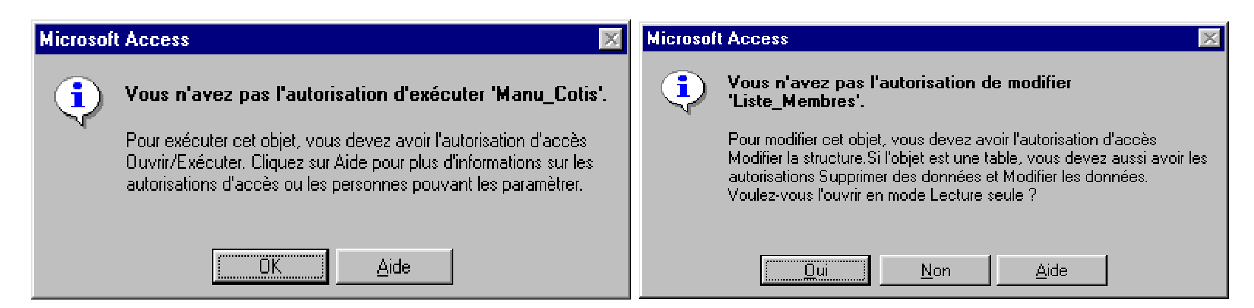
ou le même avec le **Runtime**:

F:\MsOffice\Office\Msaccess.exe G:\Ecole\Ecole.mdb /runtime /wrkgrp G:\Ecole\System.mdw /user Bigboss

ou le même mais qui détecte automatiquement le nom de l'utilisateur Windows avec les commandes batch:

F:\MsOffice\Office\Msaccess.exe G:\Ecole\Ecole.mdb /runtime /wrkgrp G:\Ecole\System.mdw /user %username%

Lorsqu'un utilisateur tente d'ouvrir, de modifier ou de détruire un objet auquel il n'a pas accès, il reçoit un message lui signifiant son éviction:

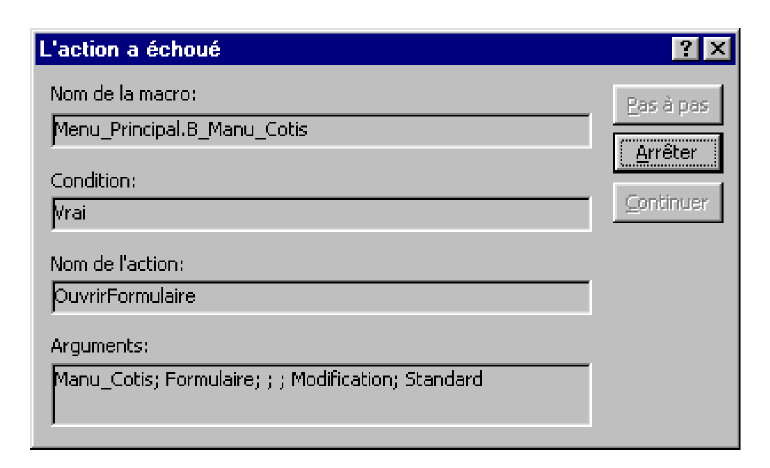


Attention!!! Si vous créez un raccourci entre une base de données et un espace de travail sécurisé comme montré précédemment et que après ouverture vous allez ensuite dans le menu *Outils/Sécurité/Administrateur du groupe de travail* vous verrez qu'il peut arriver qu'il est encore connecté peut-être à un autre fichier *.mdw (qui n'est pas celui spécifié dans le raccourci…). Dès lors, n'ayez aucune inquiétude, le bon fichier *.mdw a quand même été chargé correctement en mémoire!

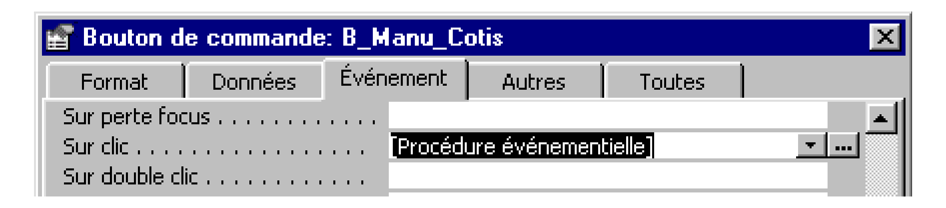
17.2.5 Un peu de VBA avec la sécurité

Cependant, lorsque un utilisateur clique sur un bouton qui le conduit dans un domaine auquel il n'a pas accès, le résultat n'est pas très heureux si l'action est mise en œuvre par une macro:

Le message apparaît …… et la macro se plante !



En conséquence, il vaut mieux être méthodique et respecter cette consigne: dans une base de données sécurisée, les actions des boutons doivent être générées par une procédure événementielle qui comporte un traitement des erreurs !



```
Private Sub B_Manu_Cotis_Click()

On Error GoTo Err_B_Manu_Cotis_Click

    DoCmd.OpenForm "MANU_COTIS"      <----  L'utilisateur a le droit
                                            d'accès :
Exit_B_Manu_Cotis_Click:                    Ouverture normale

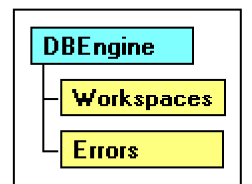
    Exit Sub

Err_B_Manu_Cotis_Click:

    MsgBox Error$                <----  Pas de droit d'accès :
    Resume Exit_B_Manu_Cotis_Click      Message d'erreur et sortie

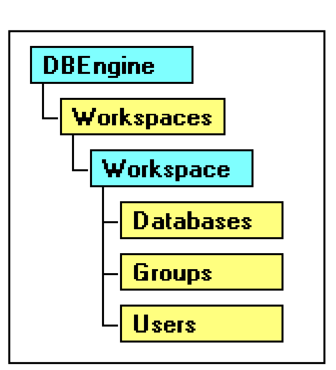
End Sub
```

L'environnement ACCESS est structuré en "Objets" et en "Collections d'objets".



L'objet originel et unique est « DBEngine », le « moteur » du système de gestion de base de données, c'est à dire l'ensemble des programmes qui contrôle tout le processus.

Ce moteur contrôle deux collections (ou ensembles) d'objets: les "Espaces de travail" et la liste des erreurs.

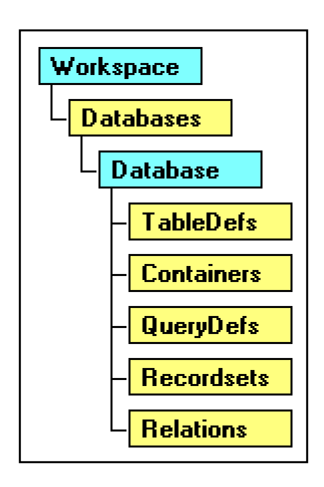


On voit donc que le terme "Espace de travail" est le mot générique et que la traduction en "Groupe de travail" est un peu confondante.

Le moteur permet de gérer les différents "Espaces de travail".

Chaque espace de travail peut contenir un certain nombre de groupes d'objets:

- Databases: une ou plusieurs Bases de données
- Groups: un ou plusieurs Groupes
- Users: un ou plusieurs Utilisateurs.



Groups et Users sont complémentaires:

Chaque groupe peut compter un ou plusieurs utilisateurs, chaque utilisateur peut faire partie de plusieurs groupes.

Pour compléter cette information, on voit qu'une Base de données est composée de collections d'objets très bien organisés:

- TableDefs: les descriptions de toutes les tables et de toutes les propriétés de tous leurs champs.
- Recordsets: tous les enregistrements de toutes les tables

- Relations: les descriptions de toutes les relations existant entre toutes les tables
- QueryDefs: les descriptions de toutes les requêtes
- Containers: les descriptions de tous les formulaires et états.

Visual Basic pour Access permet de traiter tous les objets de toutes les manières possibles: création, modification, suppression !

VBA offre une systématique remarquable pour:

- Désigner chaque objet
- Modifier ses propriétés
- Exécuter ses méthodes.

Avant de passer en revue les propriétés et les méthodes de tous ces objets, il faut tout d'abord s'occuper de deux besoins élémentaires pour la gestion des droits d'accès:

- Obtenir le nom de l'utilisateur courant
- Obtenir le groupe auquel appartient cet utilisateur.

Obtenir le nom de l'utilisateur courant:

```
Function QuelUser()
   Dim Espace As Workspace
   Set Espace = DBEngine.Workspaces(0)
   QuelUser = Espace.UserName
   Set Espace = Nothing
End Function
```

Obtenir le groupe de l'utilisateur courant (à part Admins et Users)

```
Function QuelGroupe()
   Dim Espace As Workspace
   Dim Branché As USER
   Dim LeGroupe As Group
   Set Espace = DBEngine.Workspaces(0)
   Set Branché = Espace.USERS(Espace.UserName)
   For Each LeGroupe In Branché.Groups
      If LeGroupe.Name = "Admins" Or LeGroupe.Name = "Users" Then
      Else
         QuelGroupe = LeGroupe.Name
      End If
   Next LeGroupe
   Set Espace = Nothing
   End Function
```

17.2.6 A propos du fichier ldb

Un fichier est créé dans le répertoire à chaque accès à l'un des fichiers d'une application, ce comportement est valable pour toute application quel que soit l'architecture et le nombre d'utilisateurs. Les fichiers créés portent le nom du fichier Access qui est ouvert (mdb, mde) suivi de l'extension ldb (*Lock file for Access **D**ata**B**ase*).

Par exemple: Pour le fichier *Magasin.mdb* le fichier portera le nom de *Magasing.ldb*.

Ce fichier contient des informations importantes: l'identification unique de l'utilisateur connecté et non du poste accédant à la base.

Le tableau suivant dresse les actions simplifiées opérées sur les fichiers ldb dans le cadre de l'utilisation d'une application de type "fichier serveur" par plusieurs utilisateurs.

Les fichiers applicatifs portent les noms respectifs de application.mdb (ou mde) pour la partie applicative (frontale) et données.mdb pour la partie contenant les données (dorsale).

Action	fichier ldb frontal: application.ldb	fichier ldb dorsal: données.ldb
Le premier utilisateur ouvre l'application	création du fichier, l'utilisateur est inscrit dans le fichier ldb.	aucun effet
Le premier utilisateur accède à une table	table résidente: aucun effet	table attachée: création du fichier ldb sur le fichier frontal
Un deuxième utilisateur se connecte	modification du fichier ldb (l'utilisateur est inscrit)	aucun effet
Le premier utilisateur ferme l'application	désinscription de l'utilisateur dans le fichier ldb	suppression du fichier ldb
Le deuxième utilisateur accède à une table	table résidente: aucun effet	table attachée: création du fichier ldb sur le fichier frontal
Le deuxième utilisateur n'accède plus à la table	table résidente: aucun effet	suppression du fichier ldb
Le deuxième utilisateur se déconnecte	suppression du fichier ldb	aucun effet

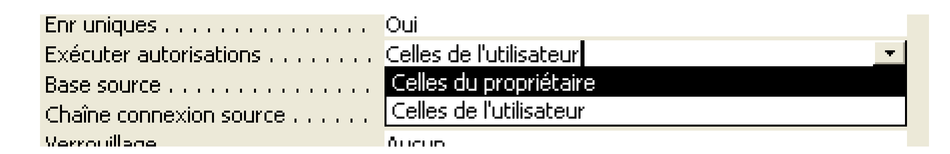
Dans de rares cas il arrive que le fichier ldb ne soit pas supprimé automatiquement par MS Access à la sortie du dernier utilisateur, dans ce cas vous pouvez le supprimer manuellement.

17.2.7 Problèmes de sécurité avec les requêtes

Lorsque la sécurité est en place, les requêtes dépendent des droits sur les tables. Pourtant certaines requêtes doivent pouvoir s'exécuter par un utilisateur ayant des droits restreints sur la ou les tables concernées. Microsoft Access permet de faire cela.

Ouvrez ou créez une requête avec le compte superutilisateur. Faites un clic droit dans la zone des tables et dans le menu contextuel choisissez *Propriétés...*

Dans *Exécuter Autorisations* choisissez *Celles du propriétaire*.



Fermez la fenêtre *Propriétés*... et sauvegardez la requête.

Il est évident que le propriétaire de la requête doit avoir les droits sur les tables concernées.

Si vous regardez le résultat de cette manipulation vous pourrez constater que la chaîne SQL comporte une option supplémentaire: *WITH OWNERACCESS OPTION*

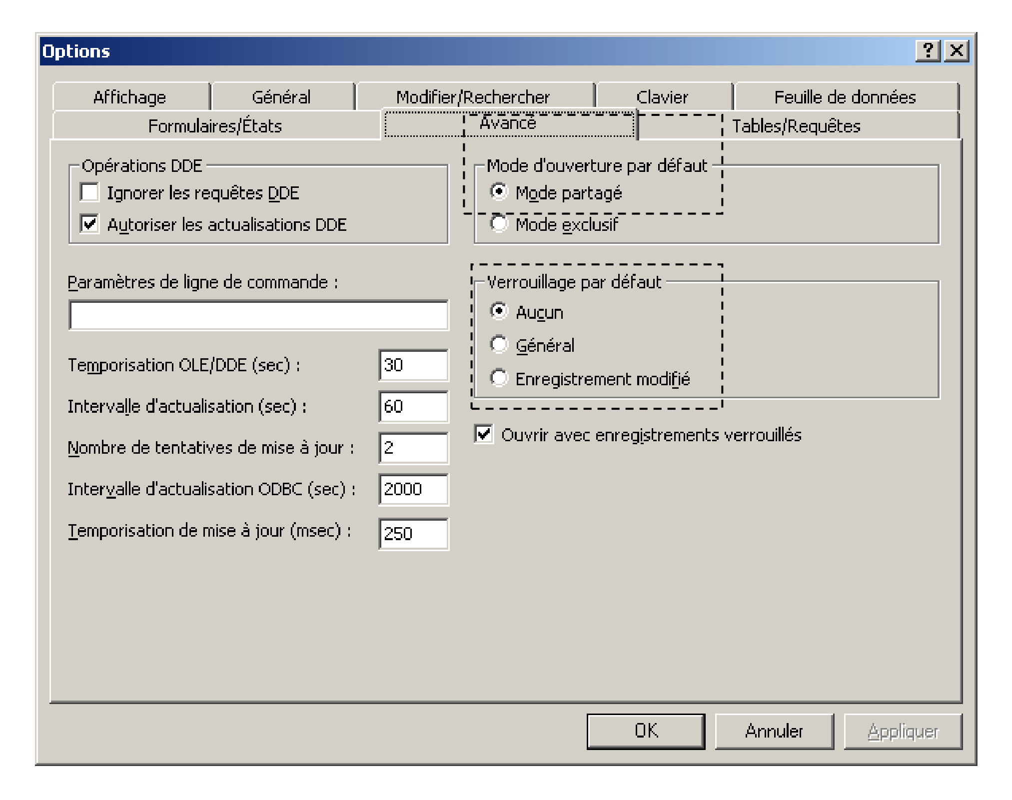
Vous pouvez utiliser cette option pour tous les types de requêtes.

17.3 Déploiement

Installez un dossier partagé sur le serveur du réseau. Peut-être aurez-vous besoin de l'administrateur de votre réseau (il peut définir des droits d'accès sur NT ce qui peut s'avérer être une couche de sécurité supplémentaire) ?

Copiez la base de données sur le serveur du réseau ou un des disques réseaux qui y sont disponibles.

Vérifiez que la base de données est définie pour une ouverture en mode partagé, qui est le paramètre par défaut (*Outils/Options/Avancé/Mode partagé*):



Pour accéder à la base de données MS Access partagée depuis un autre ordinateur, vous devez disposer sur cet ordinateur de l'une des configurations suivantes:

1. une installation locale de MS Access

2. une installation réseau de MS Access (avec licence pour chaque utilisateur) ou une application d'exécution

Vous pouvez obtenir une licence libre de droits pour installer votre application d'exécution sur tous les ordinateurs en achetant Microsoft Office 2000 Developer (MOD).

Maintenant, pour chaque participant de la salle de formation, créez un scénario d'utilisation avec droits, rôles et mots de passe respectifs. Soyez rigoureux et méthodique dans votre travail. Une fois le déploiement terminé, faites des essais et vérifiez que tout marche correctement.

Par exemple: insérez des données dans la table articles (faites jouer votre imaginaire mais tout en restant sérieux) et vérifiez que les articles de vos collègues s'y trouvent aussi.

Attention ! Vous aurez remarqué la partie suivante de la capture d'écran:

Temporisation OLE/DDE (sec) :	30
Intervalle d'actualisation (sec) :	60
Nombre de tentatives de mise à jour :	2
Intervalle d'actualisation ODBC (sec) :	2000
Temporisation de mise à jour (msec) :	250

Elle est très importante en mode multi-utilisateurs et surtout en mode serveur....

18 Synchronisation (réplication)

Nous avons vu pas mal de choses jusqu'ici mais avec la multiplication des ordinateurs portables, la mobilité croissante des employés et leur manque de connaissance des outils de l'informatique, il devient indispensable de trouver un moyen de poser la base de données sur plusieurs ordinateurs et d'avoir la possibilité de faire à tout moment une "réplication".

La réplication est une technique qui consiste à dupliquer une base de données sur plusieurs postes. Ainsi, plusieurs personnes sur le réseau peuvent chacun avoir leur copie de la base de donnée sans avoir à ouvrir le même fichier, et récupérer ou envoyer les données supprimées, ajoutée ou modifiées à l'aide d'une synchronisation avec une base commune que l'on place sur le réseau commun aux utilisateurs.

18.1 Types de réplicas

Dans le cadre du domaine de la synchronisation il convient des considérer les possibilités (fichier MDB suivants):

- Design Master: Réplica maître qui contrôle les autres réplicas. Une synchronisation depuis le Design Master permet de convertir un réplica comme étant le nouveau Design Master (l'ancien devenant alors un réplica automatiquement). Le Design Master peut créer plusieurs réplicas ou réplicas partiels (voir ci-dessous).

- Réplica: Peut se synchroniser uniquement avec son Design Master et (comme dit ci-dessus) devenir le nouveau Design Master. Le réplica peut créer aussi des sous réplicas ou réplicas partiels mais qui pourront se synchroniser qu'avec leur réplica père.

- Réplica partiel global: Peut se synchroniser avec le Design Master ou tout réplica non partiel. Un réplica partiel global ne pourra jamais devenir un Design Master. Il ne pourra également jamais créer un réplica ou un réplica partiel.

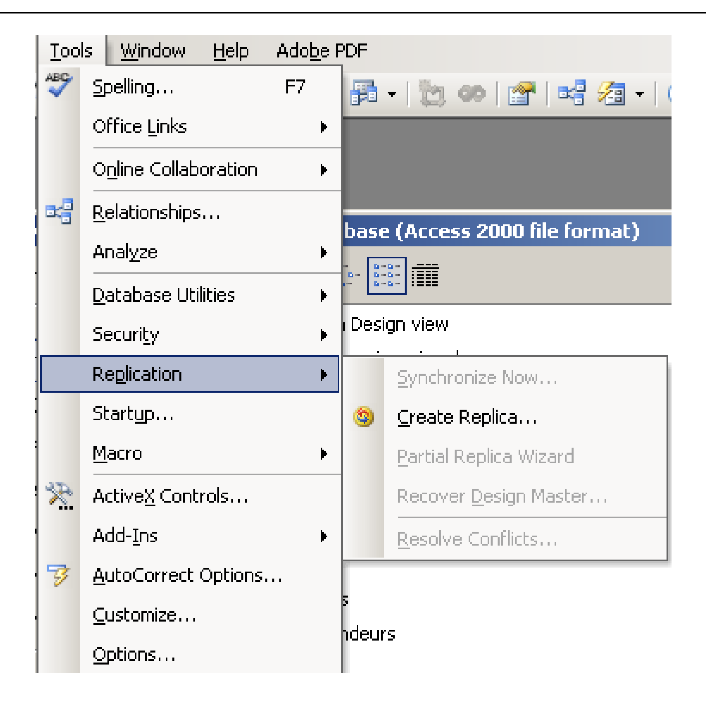
- Réplica partiel local: Peut se synchroniser seulement avec la base de données à partir de laquelle il a été créé. Un réplica partiel local ne pourra jamais devenir un Design Master. Il ne pourra également jamais créer un réplica ou un réplica partiel.

- Réplica partiel anonyme: Peut se synchroniser seulement avec la base de données à partir de laquelle il a été créé. Un réplica partiel anonyme ne pourra jamais devenir un Design Master. Il ne pourra également jamais créer un réplica ou un réplica partiel. Le réplica partiel anonyme est optimisé afin que la synchronisation utilise le minimum de flux de données possibles pour des synchronisations via des connexions web.

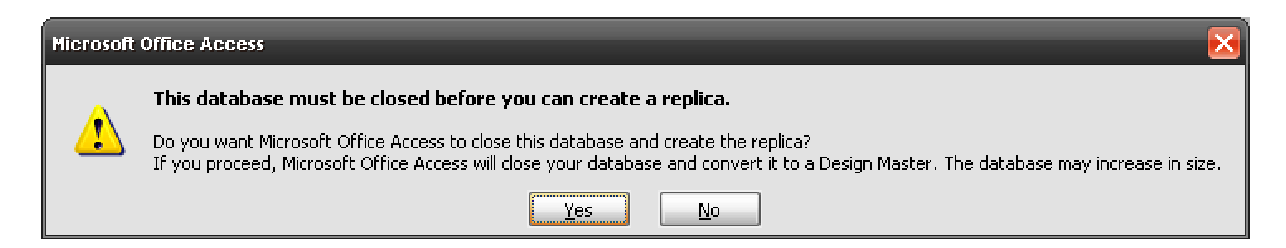
18.2 Création d'un réplica maître

Attention! Comme il n'est pas possible à ce jour d'annuler le Réplica d'une base de données sans fournir un énorme effort en termes d'heures de travail, faites toujours une copie de la base de données originale avant!

Pour utiliser cette fonctionnalité, il suffit de faire un "réplica" de la base en cours. A cette fin, allez dans le menu *Outils/Replication/Créer Replica*:



Vient alors:

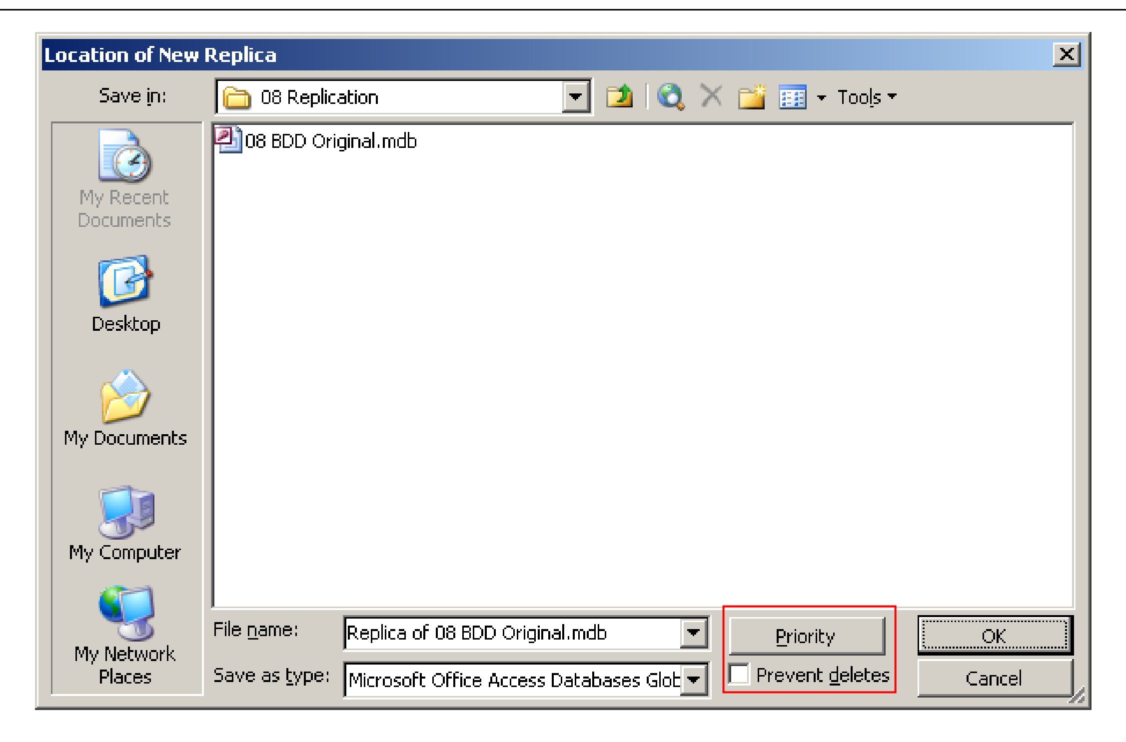


Validez pas Oui.

Ensuite, apparaît une recommandation de création d'une copie de sauvegarde (fichier *.bak) de la base initiale va apparaître. Faites-le ! Le conseil est bon !



Une fois que MS Access a avancé, la boite de dialogue ci-dessous apparaît:

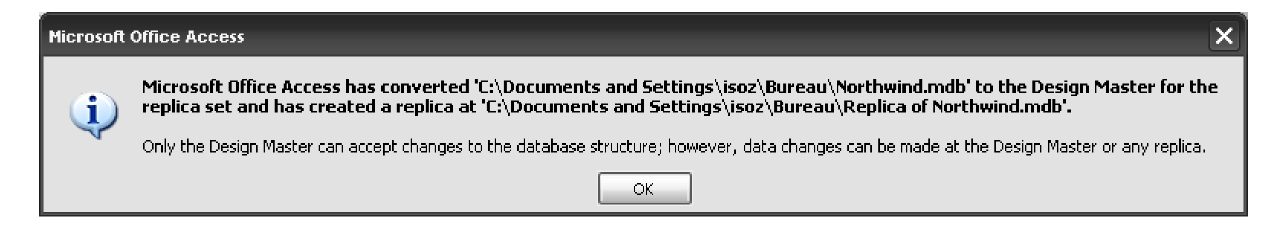


Les deux options mises en évidence sont importantes:

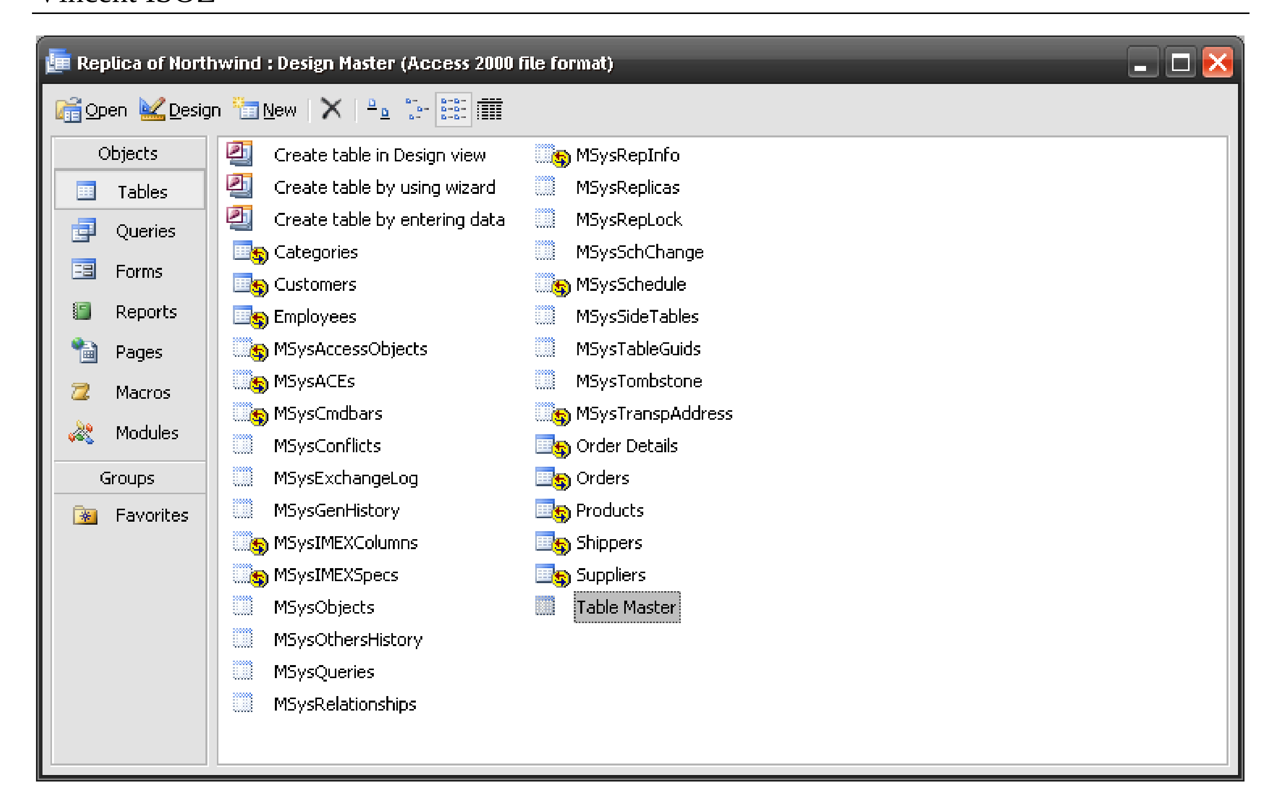
R1. *Priorité*: est une valeur de 0 à 100 qui permet de régler les conflits en cas de synchronisation simultanée d'enregistrements similaires.

R2. *Prévenir les suppressions*: empêche l'utilisateur de supprimer un enregistrement (cochez-le toujours!)

Une fois la procédure terminée vient alors:

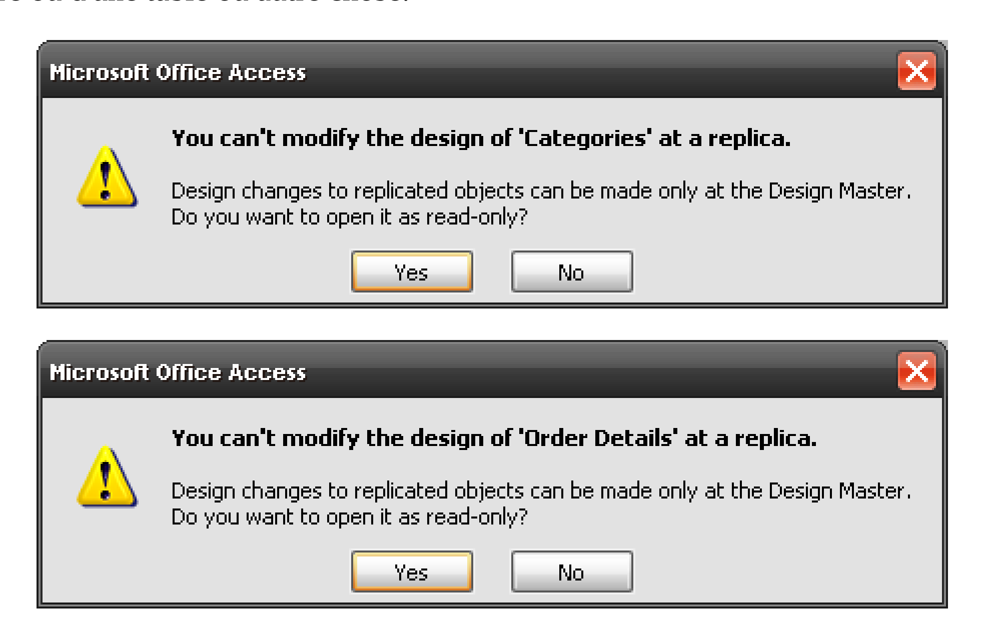


Alors, aussi bien dans le Design Master que dans le Replica il y a alors partout le symbole de la réplication:



Nous voyons également dans la capture ci-dessous, que si nous activons l'affichage des tables cachés, apparaît alors une certaine quantité de nouvelles tables permettant à MS Access de gérer la synchronisation.

Dans le réplica que ce soit dans le cadre de tentative de modification de la structure d'un formulaire ou d'une table ou autre chose:



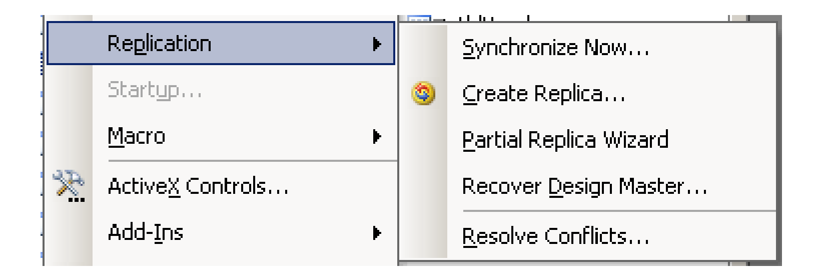
d'où le nom de Design Master pour la base de données d'origine!!!

Pour vérifier que le système de réplication fonctionne convenablement, vous pouvez apporter une modification à une donnée de votre choix (suppression, addition ou modification) dans le replica.

Si vous essayez de supprimer vous aurez (si la case à cocher de *Prévention de la suppression a bien été activée*) le message suivant:

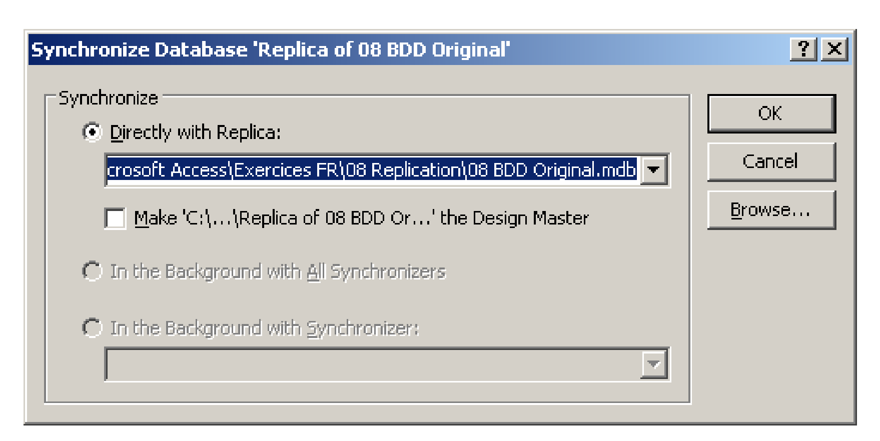


Une fois les modifications effectuées, pour synchroniser le réplica, vous retournez dans le menu *Outil* tel que:

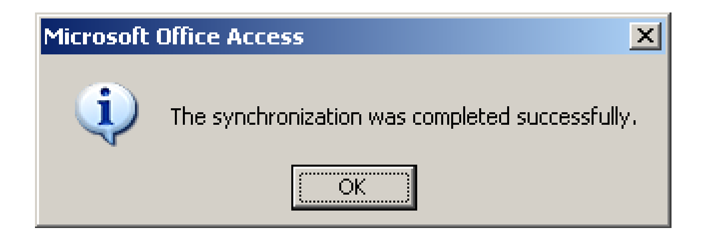


Il y a pas mal de choix et elles sont toutes triviales à utiliser. Nous allons cependant cliquer sur *Synchroniser maintenant*.

Apparaît alors la boîte de dialogue suivante vous demandant de chercher ou valider le chemin d'accès vers la base de données d'origine. Suite à quoi tout ce fait seul !



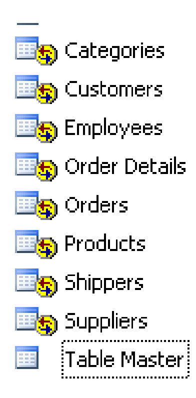
Comme une lettre à la poste….



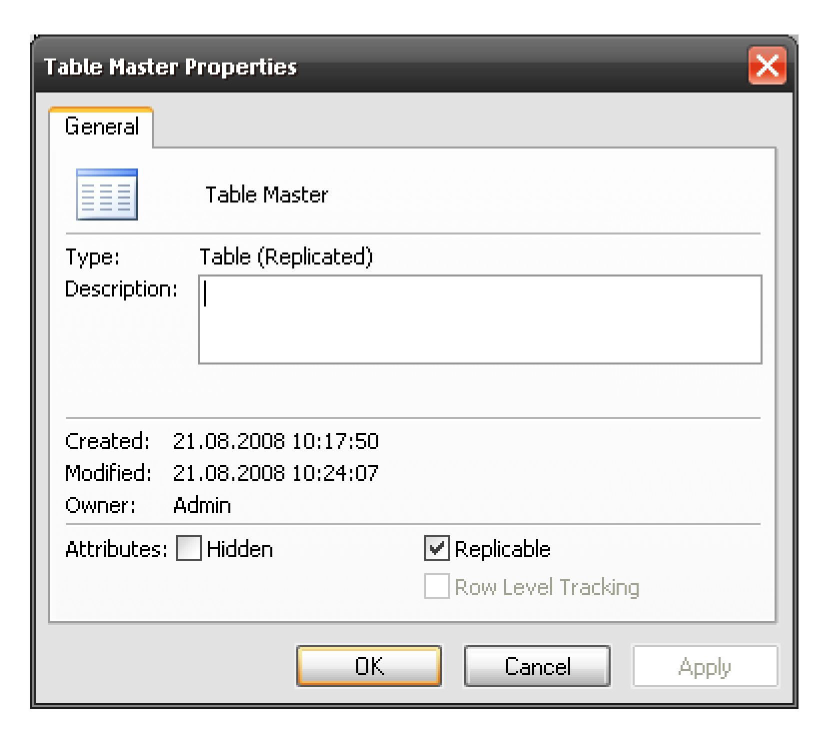
Par ailleurs, nous pouvons constater qu'une nouvelle table créée dans le réplica ne sera pas répliquée dans le Design Master:



Idem si nous en créons une dans le Design Master:



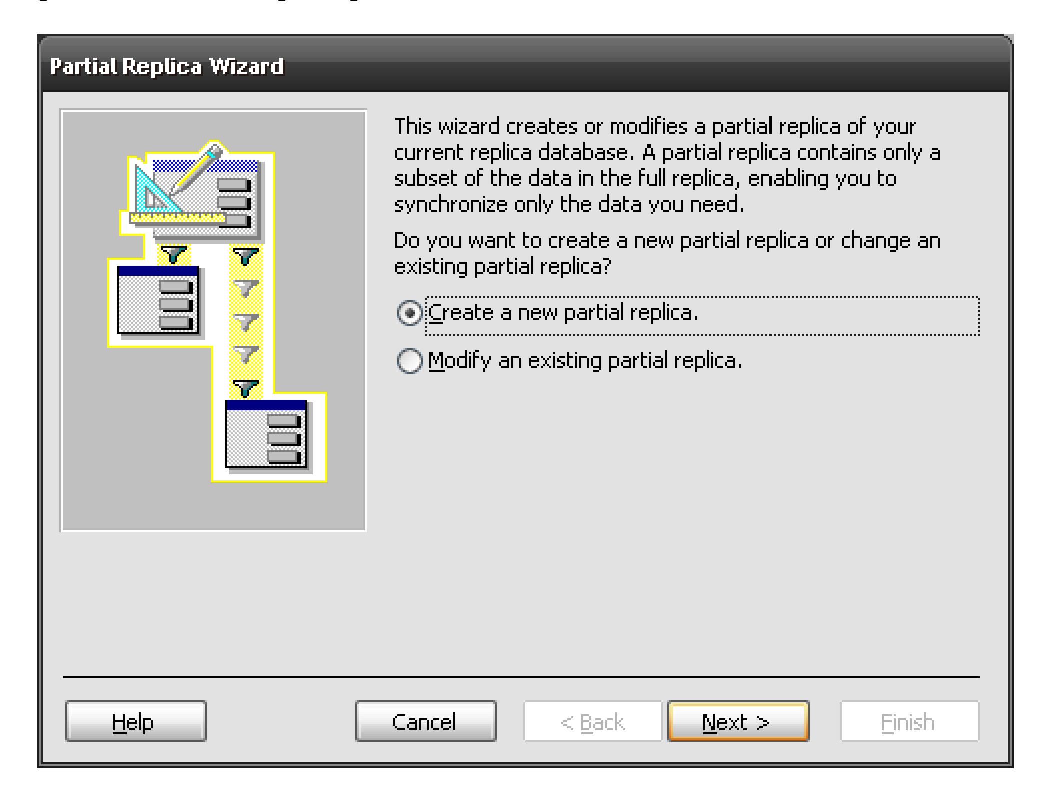
mais dans le Master on peut rendre la table réplicable en allant dans ses propriétés et en cochant l'option ad hoc:



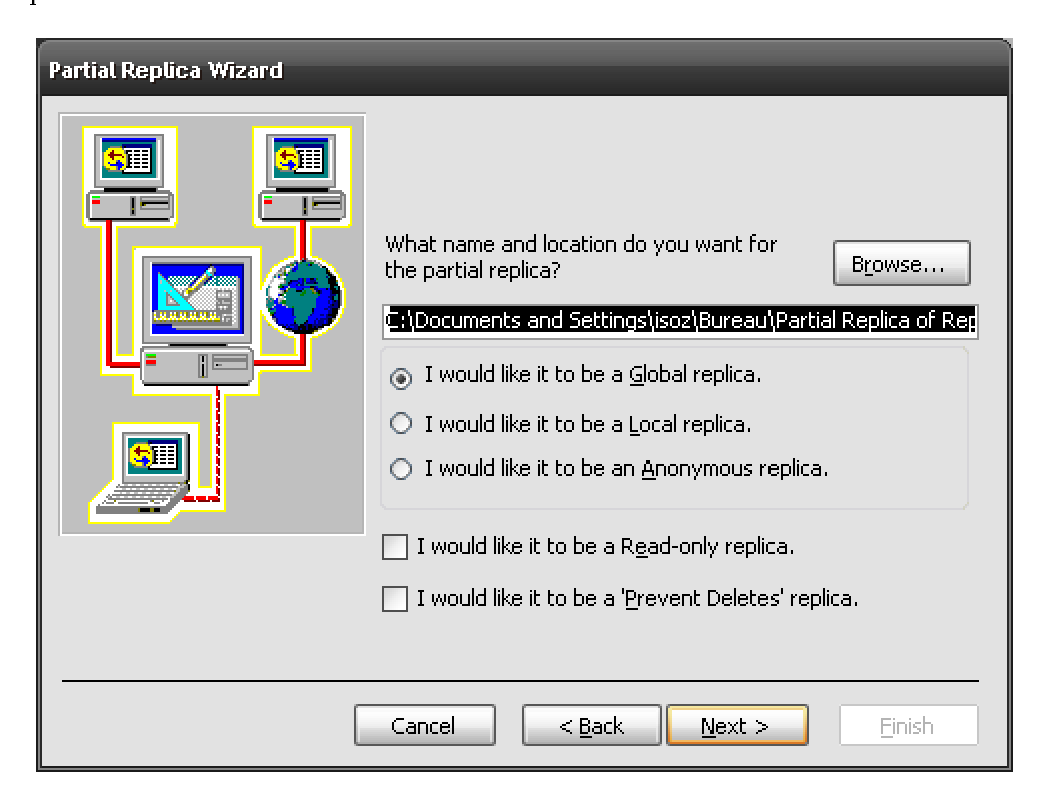
et ensuite si depuis le réplica on synchronise, cette nouvelle table sera disponible. A l'inverse si on décoche la table disparaîtra du réplica.

18.3 Création d'un réplica partiel

Pour créer un réplica partiel, il suffit toujours d'aller dans le même menu *Outils/Replication/Créer Replica partiel*. Vient alors:



et en cliquant sur Suivant:



18.4 Modifications sur les objets de la base de données

Lorsque vous répliquez une base de données Microsoft Access, un certain nombre de modifications sont effectuées automatiquement dans votre base de données.

Champs ajoutés à vos tables lorsque vous répliquez une base de données

s_GUID: Identificateur global aléatoire unique de chaque enregistrement pour assurer avec une très grand probabilité l'unicité d'un enregistrement (normalement le GUID est construit en partie sur la base de la date et l'heure de l'ordinateur).

s_Lineage: Champ binaire contenant des informations sur l'historique des modifications apportées à chaque enregistrement.

s_Generation: Champ qui stocke les informations relatives aux groupes de modifications.

Tables ajoutées à votre base de données lorsque vous répliquez celle-ci:

MSysSidetables: Cette table existe uniquement en cas de conflit entre le réplica de l'utilisateur et un autre réplica du jeu. Elle n'est pas répliquée. Elle est fournie à titre d'information seulement et son contenu ne peut être ni modifié, ni supprimé par des routines de résolution de conflit personnalisées ou par l'utilisateur. Toutes les tables latérales sont nommées table_conflit, où table est le nom d'origine de la table.

MSysSchemaProb: Cette table existe uniquement en cas d'erreur lors de la mise à jour de la structure d'un réplica. Elle offre des informations supplémentaires sur l'origine de l'erreur. Il s'agit d'une table locale, qui n'est pas répliquée.

MSysReplicas: Cette table stocke des détails, tels que le chemin et l'ID de réplica, relatifs à tous les réplicas connus contenus dans le jeu. Elle apparaît dans chaque membre du jeu de réplicas mais n'est pas répliquée.

MSysTransAddress: Cette table stocke les informations d'adressage du Synchronisateur et définit le jeu des synchronisateurs connus dans ce jeu de réplicas. Cette table répliquée apparaît dans chaque membre du jeu de réplicas.

MSysTombstone: Cette table stocke les informations relatives aux enregistrements supprimés et autorise la dispersion des suppressions dans les autres réplicas lors de la synchronisation. Elle apparaît dans chaque membre du jeu de réplicas, mais elle n'est pas répliquée.

MSysRepInfo: Cette table stocke les informations relatives à l'ensemble du jeu de réplicas, notamment l'identité (GUID) du réplica-maître. Elle contient un seul enregistrement. Cette table répliquée apparaît dans chaque membre du jeu de réplicas.

MSysExchangeLog: Cette table stocke les informations relatives aux synchronisations de réplicas qui ont été exécutées. Il s'agit d'une table locale, qui n'est pas répliquée.

Propriétés ajoutées à votre base de données lorsque vous la répliquez:

Replicable ou *ReplicableBool*: Propriété de base de données ou d'objet. Lorsque la propriété est renseignée par *V* (ou *Vrai* pour *ReplicableBool*), elle indique que la base de données, la

table ou la requête est à présent réplicable. Les propriétés *Replicable* et *ReplicableBool* sont interchangeables.

KeepLocal: Propriété ajoutée dans une table ou une requête. Si la propriété est renseignée par *V*, elle indique que l'objet ne doit pas être répliqué lorsque la base de données est répliquée. La propriété **KeepLocal** d'un objet déjà répliqué ne peut pas être renseignée par *T*.

ReplicaID: Propriété qui attribue une identification unique à chaque membre du jeu de réplicas. Cette propriété est en lecture seule et est stockée dans la table système *MSysReplicas*.

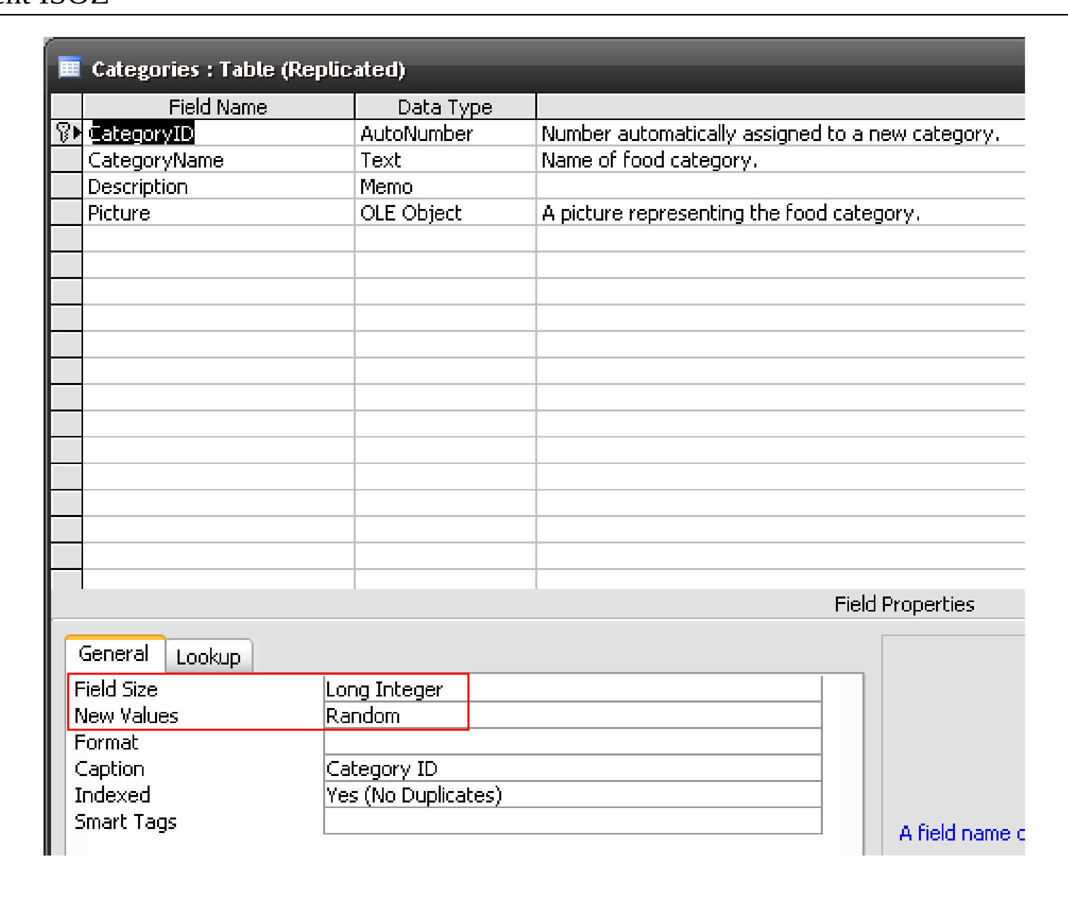
DesignMasterID: ID de réplica (*ReplicaID*) du réplica-maître. Cette propriété est stockée dans la table système *MSysRepInfo* comme *SchemaMaster*.

ColumnLevelTracking: Propriété de base de données ou de table. Lorsque cette propriété est renseignée par Vrai (valeur par défaut), elle indique que les conflits sont suivis au niveau des colonnes d'une table.

Replication ConflictFunction: Propriété qui permet de remplacer l'observateur de conflits de Microsoft Access par une procédure personnalisée qui assiste les utilisateurs dans la résolution des conflits de synchronisation.

Modifications du comportement des champs NuméroAuto en cas de réplication d'une base de données:

Lorsque vous répliquez une base de données, tous les champs *NuméroAuto* incrémentiels de vos tables deviennent des champs de numérotation aléatoire. Tous les champs *NuméroAuto* des enregistrements existants conservent leurs valeurs, mais les valeurs *NuméroAuto* des enregistrements ajoutés sont aléatoires. En d'autres termes, les numéros des enregistrements ne reflètent pas l'ordre dans lequel les enregistrements ont été ajoutés et, par conséquent, le dernier enregistrement ajouté ne présente pas forcément la valeur la plus élevée:


Categories : Table (Replicated)
Field Name
Data Type
CategoryID
AutoNumber
Number automatically assigned to a new category.
CategoryName
Text
Name of food category.
Description
Memo
Picture
OLE Object
A picture representing the food category.
Field Properties
General
Lookup
Field Size
Long Integer
New Values
Random
Format
Caption
Category ID
Indexed
Yes (No Duplicates)
Smart Tags

18.5 Comparatif Maitre-Réplica lors de la synchronisation

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

18.6 3 Conflits rencontrés lors de la synchronisation des bases

Un conflit entre réplicas est généré lorsque deux personnes changent un même enregistrement sur la base de données chacune de leur côté et synchronisent.

Pour remédier à ces conflits, Microsoft a introduit un outil de résolution de conflits qui se lancera automatiquement lors de l'ouverture de la base de données et qui vous demandera quelle version de la ligne de donnée ou se trouve le conflit vous désirez garder. Le conflit est alors résolu lorsque vous aurez choisi et validé.

Mais, lors de l'établissement de compteurs sur une table en clé primaire et que l'on réplique et synchronise, il peut arriver que deux tables se trouvent avec le même NuméroAuto mais un enregistrement différent.

Pour éviter ce problème, il vous est possible de mettre un champ numérique sur votre table en numéro de réplication. C'est un numéro unique se présentant sous la forme {00000000-0000-0000-0000-000000000000}. Il est alors impossible que 2 enregistrements présentent le même numéro.

Il existe une autre solution pour réparer les conflits de données, elle se base sur la notion de priorités de chaque réplica:

Chaque réplica se voit attribuer un numéro de priorité compris entre 0 et 100, 100 étant la priorité la plus élevée. Quand une base de données est rendue réplicable, la priorité par défaut du réplica est définie à 90. Les réplicas suivants ont une priorité par défaut égale à 90 pour cent de la priorité du concentrateur. Les priorités des réplicas locaux et anonymes sont toujours 0. Les réplicas locaux et anonymes perdent automatiquement si leurs modifications sont en conflit avec leur réplica concentrateur global. Si un réplica local ou anonyme envoie une modification qui n'est pas en conflit au concentrateur, celui-ci prend la propriété de la modification.

18.6.1 Subtilités de prédominance du maître

Si je suis dans le Maître, et que je remplace Dupon par DuponA, et que je vais dans le réplica, et que je remplace Dupon par DuponB, et que je reviens dans le maître, et que je synchronise, je ne vois pas de conflit, le maître impose sa modification au réplica, et point. Par contre, si, ensuite, je vais dans le réplica, et que je demande la synchronisation avec le maître, alors, à ce moment-là, alors qu'il paraissait gentiment s'écraser devant les modification du maître, voilà que surgit le conflit, et le réplica a donc la possibilité de revenir à son DuponB, s'il veut, et il restera ainsi différent du maître.

En conclusion, le réplica ne peut jamais avoir de prédominance sur le maître. Il doit toujours être soumis. Le maître qui effectue des modifications, et qui se synchronise avec ses réplicas n'engendre jamais de conflits, c'est toujours lui qui gagne. Mais lorsque ses modifications peuvent altérer les données des réplicas, ceux-ci se voient alors créer une table des conflits, résumant les modifications du maître. Ainsi donc, lorsque le réplica, à son tour, essaie de se synchroniser avec le maître, l'assistant des conflits démarre et propose à l'utilisateur du réplica de soit garder les données du maître, auquel cas le conflit disparait, soit de retrouver ses propres données, auquel cas la table des conflits reste en place, et le réplica n'est alors pas la copie exacte du maître.

En réalité, le seul moment où le réplica peut modifier un enregistrement avec le maître est quand il est le seul à modifier ou supprimer une donnée.

19 Visual Basic Application

Ce chapitre s'adresse à des personnes n'ayant peu ou pas d'expérience de la programmation et désireuses de développer par la suite des documents interactifs en intégrant dans les applications MS Office du code en VBA.

Remarque: La première partie de cette partie du cours est identique à celle du cours VBA MS Excel, MS Word.

Seront particulièrement concernées par ce cours:

- Les personnes avec un esprit logique et mathématique qui désirent avoir un premier contact avec le monde de la programmation.

- Les personnes ayant à automatiser des tâches sous MS Access.

- Les personnes ayant créé quelques macros et qui veulent pouvoir en comprendre le contenu et en assimiler les subtilités et voir les possibilités.

De bonnes connaissances d'un ou plusieurs des outils de la suite MS Office est souhaitable. Une approche rigoureuse de l'informatique est essentielle (génie logiciel, algorithmique, analyse numérique,…).

Pour plus d'informations sur l'algorithmique, l'histoire des langages de programmation ou la norme syntaxique habituelles de codage, veuillez-vous référer aux documents téléchargeables sur Internet ou demander à votre formateur.

Le VBA est un langage de programmation (non réellement orienté objet) utilisé par et pour les applications MS Office listées ci-dessous:

- MS Word
- MS Excel
- MS Access (voir le cours MS Access téléchargeable sur Sciences.ch)
- MS Visio
- MS Publisher
- MS Project
- MS Outlook (à partir de la version 2002… du moins facilement)
- MS FrontPage
- MS PowerPoint

Ce langage est simple d'utilisation et n'a absolument aucun commun rapport avec le langage Visual Basic .Net (nous considérons le langage Visual Basic 6 comme mort dans ce cours). La plus grosse différence étant que le VBA ne permet pas de faire ce que nous nommons des applications en "Standalone". Nous utilisons normalement les macros ou le VBA dès que les outils WYSIWYG des logiciels de la suite MS Office ne satisfont plus nos besoins.

Enfin, rappelons qu'avant d'écrire un programme quelconque, la première des choses à faire est d'éteindre son ordinateur et de réfléchir. On peut notamment se poser les questions suivantes:

- Quel est l'objectif de mon code?
- N'est-il pas plus rapide de réaliser cet objectif manuellement? (calcul du ROI)
- Cet objectif a-t-il réellement un intérêt?
- Ce programme n'existe-il pas déjà sous une autre forme?
- Ce programme est-il réalisable?
- La réalisation de ce programme n'est-elle pas trop coûteuse?

Bien évidemment, il existe de nombreux cas où vous pourrez écrire un programme sans vous poser toutes ces questions. Ainsi, quand vous voudrez rédiger un code très simple pour automatiser une tâche précise qui n'est pas complexe, vous pourrez foncer bille en tête. En revanche, dès que le projet de code devient un peu plus amitieux, il vaut vraiment mieux se poser des questions avant de commencer à écrire du code.

Ils auront travaillé avec des formulaires et manipulé des composants utilisateurs simples comme des boutons et des champs texte.

Le VBA est un langage de programmation (non objet) utilisé par et pour les applications MS Office: MS Word, MS Excel, MS Access, MS Visio, MS Publisher, MS Project, MS Outlook (à partir de la version 2002… du moins facilement), MS FrontPage, MS PowerPoint

D'autres applications ne faisant pas parties de la suite MS Office acceptent aussi le VBA et son environnement de développement (exemple: Business Objects).

Le langage est simple d'utilisation et n'a absolument aucun rapport avec le langage Visual Basic .Net (nous considérons le langage Visual Basic 6 comme mort dans ce cours). La plus grosse différence étant que le VBA ne permet pas de faire ce que l'on nomme des applications en "Standalone".

Remarques préalables:

1. Avant de commencer ce cours, il est supposé connu, les macros automatiques (pour toute la gamme de la suite MS Office), les XLA (pour MS Excel), les groupes de Macros (pour MS Access).

2. la partie débogage et la conception "objet" (je sais, je sais,… VBA n'est pas un langage POO…) ainsi que la protection des projets n'est traitée qu'oralement par le formateur en classe.

3. Nous utiliserons le VBA dès que les outils WYSIWYG des logiciels de la suite MS Office ne satisfont plus nos besoins (le problème c'est que souvent les gens ne connaissant même pas parfaitement le logiciel incriminé avant de prendre un cours VBA). L'inconvénient d'une formation VBA, c'est que autant vous pouvez effectuer une formation de niveau moyen sur MS Access, Excel ou Word sur 7 jours à 8 heures par jour et avoir vu 90% des fonctionnalités WYSIWYG, autant en ce même laps de temps, vous verrez 10% (et encore!!!) des possibilités du VBA dans chacun de ces logiciels.

19.1 Objectifs

A la fin de ce chapitre, les participants sauront parfaitement utiliser les macros automatiques et macro complémentaires et connaîtront les notions et structures standards de la programmation VBA, telles que les variables, les boucles, les conditions et les fonctions.

Ils sauront ce que sont la programmation objet et la programmation évènementielle et auront réalisés quelques exercices utilisant ces notions.

Ils auront travaillé avec des formulaires et manipulé des composants utilisateurs simples comme des boutons et des champs texte.

Remarque: On sait que le nombre de mots d'une langue est limité. Le vocabulaire d'un enfant de 10 ans tourne autour de 5'000 mots, celui d'un adulte cultivé de 10'000-15'000, et les dictionnaires en plusieurs volumes peuvent monter de 130 000 à 200 000. Le VBA d'après de rumeurs contiendrait environ 800'000 mots… (à vérifier quand même!).

19.2 Historique

En programmation, **BASIC** est un acronyme pour **Beginner's All-purpose Symbolic Instruction Code** qui désigne une famille de langages de programmations de haut niveau.

Le BASIC a été conçu à la base en 1963 par John George Kemeny (1926-1993) et Thomas Eugene Kurtz (1928-) au Dartmouth College pour permettre aux étudiants qui ne travaillaient pas dans des filières scientifiques d'utiliser les ordinateurs et apprendre les techniques de programmation. En effet, à l'époque, l'utilisation des ordinateurs nécessitait l'emploi d'un langage de programmation réputé réservé aux seuls les spécialistes, en général un langage d'assemblage ou Fortran.

L'acronyme BASIC est lié au titre d'un article de Thomas Kurtz qui n'a pas été publié et n'a aucun rapport avec les séries intitulées « Anglais basic » de C. K. Ogden. Les concepteurs du langage souhaitaient qu'il soit du domaine public, ce qui favorisa sa diffusion.

Le BASIC est indissociable de l'apparition, dans les années 1980, de la micro-informatique grand public. En effet, la plupart des micro-ordinateurs vendus durant cette période étaient fournis avec un Interprète BASIC, et quelques calculatrices programmables en furent même dotées.

Ce n'était jamais l'intention des créateurs du langage qu'il s'agit d'un langage professionnel. Pourtant il s'est propagé rapidement et est disponible dans les centaines de dialectes sur de nombreux types d'ordinateurs. La BASIC a évolué et s'est amélioré au cours des années. À l'origine, c'était un langage interprété (chaque ligne était interprétée avant son exécution… ce qui est le cas du VBA par exemple…) qui impliquait une exécution lente. La plupart des dialectes modernes de BASIC permettent au code d'être compilé. Par conséquent, l'exécution est beaucoup plus rapide et la portabilité des programmes améliorée.

Les huit principes de conception du BASIC étaient:

1. Être facile d'utilisation pour les débutant(e)s (Beginner)

2. Être un langage généraliste (All-purpose)

3. Autoriser l'ajout de fonctionnalités pour les expert(e)s (tout en gardant le langage simple pour les débutant(e)s)

4. Être interactif

5. Fournir des messages d'erreur clairs et conviviaux

6. Avoir un délai de réaction faible pour les petits programmes

7. Ne pas nécessiter la compréhension du matériel de l'ordinateur

8. Isoler (shield) l'utilisateur du système d'exploitation

Le BASIC a gagné sa respectabilité en 1991 lorsque Microsoft a lancé VISUAL BASIC pour MS Windows. Ce produit a été très populaire parmi les développeurs d'applications autonomes. Si VBA ressemble peu à ces langages, le BASIC reste la base sur laquelle VBA a été élaboré.

EXCEL 5 a été la première application sur le marché à proposer VBA et il est maintenant inclus dans presque toutes les applications de la suite bureautique depuis MS Office 97 et même chez d'autres fournisseurs. Par conséquent, si vous maîtrisez l'utilisation de VBA, vous pouvez écrire des macros avec toutes sortes d'applications (Microsoft et autres).

Il est important de noter l'information suivante (capture d'écran du site web de Microsoft):

Visual Basic for Applications

Discontinuation of the VBA Licensing Program

Since June 1996, when we first announced the Microsoft® Visual Basic® for Applications (VBA) licensing program, we have been offering VBA for licensing to Independent Software Vendors and others who wished to integrate VBA into their own applications. As previously announced, Microsoft does not expect to make significant enhancements to VBA. This does not impact the current support commitments for VBA in any way, and of course, it does not impact any license arrangements that are in force. In particular, this does not impact VBA in Microsoft Office products.

Microsoft is investing its application programmability resources in Microsoft® Visual Studio® Tools for Applications (VSTA) and its companion set of tools, Microsoft® Visual Studio® Tools for Office (VSTO). We encourage you to consider VSTA for new applications that require application programmability technology. Summit Software is Microsoft's vendor for VSTA licensing.

As of July 1, 2007, Microsoft will no longer offer VBA distribution licenses to new customers. Existing VBA customers can still purchase additional VBA licenses from Summit Software and Microsoft for existing solutions.

19.3 Types de données

Voici les types de données communes aux logiciels de la suite MS Office:

Le tableau suivant présente les types de données reconnus en précisant la taille des enregistrements et la plage des valeurs.

Tableau 8 Type de données VBA

Type de données	Taille d'enregistrement	Plage

Type de données	Taille d'enregistrement	Plage
Byte	1 octet	0 à 255
Boolean	2 octets	True ou False
Integer	2 octets	-32 768 à 32 767
Long (entier long)	4 octets	-2 147 483 648 à 2 147 483 647
Single (à virgule flottante en simple précision)	4 octets	-3,402823E38 à -1,401298E-45 pour les valeurs négatives ; 1,401298E-45 à 3,402823E38 pour les valeurs positives
Double (à virgule flottante en double précision)	8 octets	-1,79769313486231E308 à -4,94065645841247E-324 pour les valeurs négatives ; 4,94065645841247E-324 à 1,79769313486232E308 pour les valeurs positives
Currency (entier à décalage)	8 octets	-922 337 203 685 477,5808 à 922 337 203 685 477,5807
Decimal	14 octets	+/- 79 228 162 514 264 337 593 543 950 335 sans séparateur décimal ; +/-7,9228162514264337593543950335 avec 28 chiffres à droite du séparateur décimal ; le plus petit nombre différent de zéro est +/- 0.0000000000000000000000000001.
Date	8 octets	1er janvier 100 au 31 décembre 9999
Object	4 octets	Toute référence à des données de type Object
String (longueur variable)	10 octets + longueur de la chaîne	0 à environ 2 milliards
String (longueur fixe)	Longueur de la chaîne	1 à environ 65 400
Variant (nombres)	16 octets	Toute valeur numérique, avec la même plage de valeurs qu'une donnée de type Double
Variant (caractères)	22 octets + longueur de la chaîne	Même plage de valeurs qu'une donnée de type String de longueur variable
Type défini par l'utilisateur (avec Type)	En fonction des éléments	La plage de valeurs de chaque élément correspond à celle de son type de données.

Remarque: Quel que soit le type de données, les tableaux nécessitent 20 octets de mémoire, auxquels viennent s'ajouter quatre octets pour chaque dimension et le nombre d'octets occupés par les données. L'espace occupé en mémoire par les données peut être calculé en multipliant le nombre d'éléments par la taille de chacun d'eux. Par exemple, les données stockées dans un tableau unidimensionnel constitué de quatre éléments de type Integer de deux octets chacun occupent huit octets. Ajoutés aux 24 octets d'espace mémoire de base, ces huit octets de données portent la mémoire totale nécessaire pour le tableau à 32 octets.

Une variable de type Variant contenant un tableau nécessite 12 octets de plus qu'un tableau seul.

Remarque: Utilisez la fonction StrConv pour convertir un type de données de chaîne en un autre

L'existence d'une variable peut se dérouler sur trois niveaux:

1. **Niveau Procédure:** cela veut dire que la variable est locale. Dès que l'on quitte la procédure en question, la variable disparaît, et son contenu avec elle. Pour déclarer une variable au niveau procédure, on tape à l'intérieur de la procédure:

   ```
   Dim NomVariable as Type
   ```

2. **Niveau Module**: la variable est disponible pour toutes les procédures d'un Module, mais pas pour les procédures se situant sur un autre Module. Pour déclarer une variable au niveau Module, on tape tout en haut du Module, dans la partie (General):

   ```
   Private NomVariable as Type
   ```

3. **Niveau Projet**: la variable est disponible, et sa valeur est conservée pour toutes les procédures de l'application, quel que soit leur emplacement. Pour déclarer une variable globale, il faut d'abord créer un module. Sur ce module, donc, on écrit:

   ```
   Public NomVariable as Type
   ```

Naturellement, il ne faut pas raisonner en termes de facilité, et déclarer toutes les variables au niveau projet: car l'excès de place mémoire, ralentira votre application, au besoin considérablement. Il faut donc pour chaque variable se demander à quel niveau on en a besoin, et faire les bonnes déclarations en fonction.

L'existence d'une procédure peut se dérouler quant à elle que sur deux niveaux:

1. **Niveau Module:** une procédure privée ne pourra être invoquée que dans le module dans lequel elle est déclarée:

   ```
   Private Sub NomProcédure()
               …
   End Sub
   ```

2. **Niveau Projet:** une procédure publique peut être invoquée de n'importe quel endroit du projet.

   ```
   Public Sub NomProcédure()
   ```

…

End Sub

19.4 Nomenclature de Lezsynski-Reddick

Pour le développement (base de données et autres), il y a certaines règles et traditions qu'il vous faut respecter dans un premier temps pour votre confort et dans un deuxième temps pour être compatible avec vos collègues et les possibles futures migrations.

Les règles Lezsynski/Reddick© pour le développement de base de données sont les suivantes (elles ont été également adoptées pour d'autres langages de programmation):

Majuscule au début de chaque mot d'une variable, pas d'accents, pas de caractères spéciaux, pas d'espaces, nom des champs en anglais, éviter de dépasser les 8 caractères, ne jamais commencer avec des chiffres:

- Nom des tables: tbl….

- Nom des requêtes: qry…

- Nom des vues: vue…

- Nom des états: rep…

- Nom des formulaires: frm…

- Nom des champs clés primaire avec numéro automatique: idNomTable

- Nom des champs clés étrangères: tblNomTableNomChamp

- Nom de tous les autres champs:

strNom…, intNom…, datNom…, oleNom…, hypNom…, bolNom…

- Nom des champs de formulaire:

lstNomListe…., optGroupeOptions…., chkChoixCase…., tglToggleButton…, fldNomChamp…

Exemple d'une variable: *intStreetNb.* Ce qui est beaucoup mieux que *Numéro de la rue* qui ne nous donne pas d'un premier coup d'œil, le type de données dont il s'agit, qui n'est pas compatible avec la quasi-totalité des langages de programmation, et qui peut être compris par un maximum de personne de par l'usage de la langue anglaise.

Il est aussi possible d'utiliser une version condensée de la norme ci-dessous connue sous le nom "syntaxe Camel" utilisée par la majorité des développeurs (seniors).

Préfixes de variables:

Préfixe	Emploi de la variable	Exemple de variable
b ou bln	Booléen	bSuccess
c ou cur	Monnaie	cAmount

d ou dbl	Double	dblQuantity
dt ou dat	Date et heure	dtDate
f ou flt	Flottant	fRatio
l ou lng	Long	lMilliseconds
i ou int	Entier	iCounter
s ou str	Chaîne	sName
a ou arr	Tableau	aUsers()
o ou obj	Objet COM	oPipeline

Préfixes de variables pour les objets de base de données:

Préfixe	Emploi de la variable	Exemple de variable
cnn	Connexion	cnnPubs
rst	Jeu d'enregistrements	rstAuthors
cmd	Commande	cmdEmployee
fld	Champ	fldLastName

Préfixes d'étendue et d'usage:

Préfixe	Description
g_	Usage Public
m_	Usage Local
(pas de préfixe)	Variable non statique, préfixe local à la procédure

Les six règles d'or d'usage:

R1. Toujours en anglais

R2. Entre 8 et 11 caractères

R3. Pas de caractères spéciaux

R4. Pas d'espace

R5. Toujours les 3 premières lettres du type de données

R6. Une majuscule à chaque premier lettre des mots des variables

19.5 Commentaires

Les commentaires du code est un point très important du développement que l'on comprend seulement avec l'expérience. Effectivement, les développeurs débutants n'ont dans un premier temps pas l'habitude de travailler très rigoureusement et très proprement et ce d'autant plus sur des codes rarement supérieur à 1000 lignes et ne voient donc pas quelle est l'intérêt futur de bien commenter leur code. Cet état des faits a lieu chez la grand majorité des développeurs.

Ne pas commenter est une énorme erreur pour le développeur lui-même et tous ceux qui seraient amenés à intervenir ou à poursuivre son travail.

Certaines règles sont à mettre en place il convient immédiatement de mettre en pratique dès que l'on commence à rédiger un code. Voici ces règles:

Tout procédure, fonction, classe, doit être accompagnée d'une cartouche de description telle que dans l'exemple ci-dessous

Chaque ligne de code doit être commentée avec indication en initiales du commentateur et de la date de création du commentaire tel que dans l'exemple ci-dessous

Au besoin, un schéma procédural doit être fait dans un logiciel adapté (MS Visio pour VBA suffit) pendant le travail afin de savoir qui appelle quoi en faisant usage de quelles variables

Exemple de code:

```
'********************************
'Créateur(s): Vincent Isoz
'Dernière modification: 18.09.2004
'Nom fonction: TestDeVariable()
'Appelée par: -
'Commentaires: exemple de danger de conversion de données
'********************************
Dim sng As Single 'Nombre réel simple précision
Dim dbl As Double 'Nombre réel double précision
Sub SigngleToDouble()
    'on affecte 1.9 a la variable sng
    sng = 1.9
    'on affecte la valeur de sng a dbl
    dbl = sng
    'on Affiche dbl
    MsgBox dbl
    'ou encore pour les sceptique
    dbl=Cdbl(sng)
    MsgBox dbl
End Sub

'********************************
'Créateur(s): Vincent Isoz
'Dernière modification: 28.10.2003
'Nom fonction: factitfor()
'Appelée par: mettre ici les nom de procédures (avec les modules) qui appellent la fonction
'Appelle: mettre ici le nom des de procédures (avec les modules) qui sont appelé par la
fonction
'Commentaires: Calcul de la factorielle d'un nombre n par la méthode itérative "for"
'Objectif de cours: apprendre a créer des fonction itératives
'********************************
Function factitfor(n)
'V.I.(28.10.03): On ne déclare pas la variable factit qui a le même nom que la fonction!!
Dim i As Integer
   factitfor = 1
```

```
    'V.I.(28.10.03): On utilise la méthode itérative classique vue à l'école primaire
    'Attention, n et i doivent être des variables du même type !
    For i = 1 To n
        factitfor = factitfor * i
    Next i
End Function
```

19.6 Table des objets VBA et table ASCII

Outre les variables et les procédures, nous allons retrouver quantité d'autres objets à traver l'explorateur d'objets de VBAE (pour plus de détails voir le cours VBA MS Excel disponible en PDF lui aussi):

Tableau 9 Objets VBA

Les icônes de l'Explorateur d'objets			
Icône	Signification	Icône	Signification
	Type personnalisé On retrouve le même icône dans l'intellisense: équivaut à un type utilisateur: (User Defined): TYPE … END TYPE		*Propriété standard* Propriété par défaut: ex.: Label = "texte" équivaut à Label.caption="texte" N'existe plus en VB .NET
	Projet Icône Projet		Méthode
	Classe		Méthode standard Méthode par défaut
	Module Icone Module		Enumération Ensemble de constants énumérée (voir dessous)
	Globale Membre appartenant à tous les sous membres en principe les constantes		Constante Valeur nommée: Ex: vbBlack = 0 Avantage si la valeur est modifiée le code continue à fonctionner
	Evénement		Propriété

Table ASCII standard (codes de caractères de 0 … 127):

Tableau 10 Tableau ASCII Standard

000	(nul)	016	(dle)	032	(sp)	048	0	064	@	080	P	096	`	112	p
001	(soh)	017	(dc1)	033	!	049	1	065	A	081	Q	097	a	113	q
002	(stx)	018	(dc2)	034	"	050	2	066	B	082	R	098	b	114	r
003	(etx)	019	(dc3)	035	#	051	3	067	C	083	S	099	c	115	s
004	(eot)	020	(dc4)	036	$	052	4	068	D	084	T	100	d	116	t
005	(enq)	021	(nak)	037	%	053	5	069	E	085	U	101	e	117	u
006	(ack)	022	(syn)	038	&	054	6	070	F	086	V	102	f	118	v
007	(bel)	023	(etb)	039	'	055	7	071	G	087	W	103	g	119	w
008	(bs)	024	(can)	040	(	056	8	072	H	088	X	104	h	120	x
009	(tab)	025	(em)	041	)	057	9	073	I	089	Y	105	i	121	y
010	(lf)	026	(eof)	042	*	058	:	074	J	090	Z	106	j	122	z
011	(vt)	027	(esc)	043	+	059	;	075	K	091	[	107	k	123	{
012	(np)	028	(fs)	044	,	060	<	076	L	092	\	108	l	124	|
013	(cr)	029	(gs)	045	-	061	=	077	M	093	]	109	m	125	}
014	(so)	030	(rs)	046	.	062	>	078	N	094	^	110	n	126	~
015	(si)	031	(us)	047	/	063	?	079	O	095	_	111	o	127	(127)

Table ASCII étendue (codes de caractères de 128 … 255):

Tableau 11 Tableau ASCII étendu

128	€	144	•	160		176	°	192	À	208	Ð	224	à	240	ð
129	•	145	'	161	¡	177	±	193	Á	209	Ñ	225	á	241	ñ
130	‚	146	'	162	¢	178	²	194	Â	210	Ò	226	â	242	ò
131	ƒ	147	"	163	£	179	³	195	Ã	211	Ó	227	ã	243	ó
132	„	148	"	164	¤	180	´	196	Ä	212	Ô	228	ä	244	ô
133	…	149	•	165	¥	181	µ	197	Å	213	Õ	229	å	245	õ
134	†	150	–	166	¦	182	¶	198	Æ	214	Ö	230	æ	246	ö
135	‡	151	—	167	§	183	·	199	Ç	215	×	231	ç	247	÷
136	ˆ	152	˜	168	¨	184	¸	200	È	216	Ø	232	è	248	ø
137	‰	153	™	169	©	185	¹	201	É	217	Ù	233	é	249	ù
138	Š	154	š	170	ª	186	º	202	Ê	218	Ú	234	ê	250	ú
139	‹	155	›	171	"	187	"	203	Ë	219	Û	235	ë	251	û
140	Œ	156	œ	172	¬	188	¼	204	Ì	220	Ü	236	ì	252	ü
141	•	157	•	173		189	½	205	Í	221	Ý	237	í	253	ý
142	Ž	158	ž	174	®	190	¾	206	Î	222	Þ	238	î	254	þ
143	•	159	Ÿ	175	¯	191	¿	207	Ï	223	ß	239	ï	255	

Objets MS Access:

Tableau 12 Objets MS Access

Constante	Valeur entière
acTable	0
acQuery	1
acForm	2
acReport	3
acMacro	4
acModule	5
acDataAccessPage	6
acServerView	7
acDiagram	8
acStoredProcedure	9

Objet MS Access en mode ouverture:

Tableau 13 Objets MS Access en mode ouverture

Constante	Valeur entière	Support
acCurViewDesign	0	Formulaires, états, pages, macros et modules
acCurViewFormBrowse	1	Formulaires ouverts en mode Formulaire
acCurViewDatasheet	2	Formulaires ouverts en mode Feuille de données
acCurViewPivotTable	3	Formulaires ouverts en mode Tableau croisé dynamique
acCurViewPivotChart	4	Formulaires ouverts en mode Graphique croisé dynamique
acCurViewPreview	5	États ouverts en mode Aperçu avant impression

Comme vous avez pu vous en apercevoir jusqu'à maintenant, MS Access est un outil fort complet dont la combinaison de l'ensemble des possibilités fait de son utilisation un outil de métier.

Cependant même si nous avons vu quantités de choses jusqu'à maintenant, il est possible que vous trouviez que MS Access manque d'outils spécifiques à vos besoins (qui sont comme c'est très souvent le cas: particuliers!).

Nous utilise dès lors le VBA (Visual Basic Application) qui est un langage de programmation évolué dit abusivement orientée objet (ça y ressemble et cela en a presque les possibilités). Cependant, l'inconvénient, c'est que autant vous pouvez effectuer une formation en MS Access sur 7 jours à 8 heures par jour et avoir vu 90% des fonctions, autant en ce même laps de temps, vous verrez 10% (et encore!!!) des possibilités du VBA d'Access.

Remarque: précisons que MS Access VBA utilise aussi un autre langage pour fonctionner qui est le langage SQL (Stuctured Query Language) qui en lui-même fait l'objet d'un cours d'au moins 5 jours.

Avant de commencer à passer à des exercices pratiques, il est nécessaire de voir les bases élémentaires de l'algorithmique afin de savoir ce qu'est une structure conditionnelle, une itération, la récursivité, les booléens et autres points importants du domaine de la programmation…

Nous allonsa également créer des modules VBA afin de se faire la main pour voir la structure du langage et la façon dont on écrit des routines et fonctions simples (appelées avec arguments passés en paramètres et les fonctions récursives et plein d'autres choses).

Nous allons définir les notions de *méthodes*, *propriétés*, *événements* et les boîtes de dialogue. Nous allons également apprendre à utiliser *l'explorateur d'objets*, l'aide, le *débuguer* (la *fenêtre d'exécution*, la *fenêtre des variables locales*, les *espions*, les *points d'arrêt*, *l'exécution pas à pas*, la *mise en commentaires*), la *gestion des erreurs* (la commande *GoTo*).

Le lecteur se reportera ici au cours PDF MS Excel VBA disponible gratuitement au format PDF au même endroit qu'il a trouvé le présent document…

Donc Maintenant avec votre formateur voyons:

- L'interface Visual Basic Application Editor (abrégé VBAE par la suite)

- Explorateur de projets, propriétés, protection du code

19.7 Prise en main du V.B.A

Avant de commencer à coder, il est nécessaire de passer par des exemples génériques du VBA qui permettront au participant d'acquérir les bases du vocabulaire et de la grammaire du langage ainsi que les notions d'algorithme, de procédure, fonction, itération, test logique, gestion des erreurs etc.

C'est un passage obligé pour quiconque veut se prétendre faire un code un minimum acceptable…

Quelques rappels avant de voir ces exemples génériques:

R1. Le terme Option Explicit au début d'un module permet d'obliger la déclaration des variables dans le VBA (et VB). Ceci permet d'avoir une rigueur supérieure dans votre travail et peut éviter des problèmes de déclaration.

R2. Le terme Option Compare Database au début d'un module permet de traiter les caractères contenus dans les tables.

R3. L'instruction Option Compare définit la méthode de comparaison de chaînes (Binary, Text ou Database) pour un module. Si le module ne contient pas d'instruction Option Compare, la méthode de comparaison de texte par défaut est Binary.

R4. L'instruction Option Compare Binary fournit des comparaisons de chaînes basées sur un ordre de tri dérivé de la représentation binaire interne des caractères. Dans MS Windows,

l'ordre de tri est déterminé par la page de code. L'exemple suivant décrit un ordre de tri binaire typique:

A < B < E < Z < a < b < e < z < À < Ê < Ø < à < ê < ø

R5. L'instruction Option Compare Text fournit des comparaisons de chaînes basées sur un ordre de tri qui ne distingue pas les majuscules des minuscules et qui est déterminé par les paramètres régionaux de votre système. Si les caractères ci-dessus sont triés à l'aide de l'instruction Option Compare Text, l'ordre de tri de texte suivant est utilisé:

(A=a) < (À=à) < (B=b) < (E=e) < (Ê=ê) < (Z=z) < (Ø=ø)

Activer la bonne option est très utile lorsque vous développez un système de gestion des mots de passe et que vous souhaitez comparer un mot de passe saisi à un mot de passe stocké. Par exemple, le code ci-dessous ne fera pas la différence entre les deux chaînes de caractères (peu importe quelle soit leur provenance) si l'on écrit *toto* comme mot de passe:

```
Option Compare Database
Option Explicit

Sub ComparaisonMotDePasse()

   Dim strPassReference, strPasseSaisi As String

   strPassReference = "Toto"
   strPasseSaisi = InputBox("Merci de saisir votre mot de passe")

   If strPasseSaisi = strPassReference Then
      MsgBox "Mot de Passe OK", vbInformation
   Else
      MsgBox "Mot de passe pas OK", vbCritical
   End If

End Sub
```

Alors que le code suivane fera la différence des majuscules et minuscules entre *toto* et *Toto*:

```
Option Compare Binary
Option Explicit

Sub ComparaisonMotDePasse()

   Dim strPassReference, strPasseSaisi As String

   strPassReference = "Toto"
   strPasseSaisi = InputBox("Merci de saisir votre mot de passe")

   If strPasseSaisi = strPassReference Then
      MsgBox "Mot de Passe OK", vbInformation
   Else
```

```
      MsgBox "Mot de passe pas OK", vbCritical
   End If

End Sub
```

Faisons maintenant quelques exemples de codes pures et "inutiles" dans MS Access mais utiles pour l'apprentissage de la saisie de code et pour se "faire la main" (il faut toujours commencer par la base):

19.7.1 Exemples génériques à Office

```
Option Compare Database
Option Explicit

Sub factorielle()
Dim n As Integer
Dim cible As Integer
Dim resp As Byte
2    n = InputBox("Valeur de n?")
   On Error GoTo 1
   init = n
   If n = 0 Then
      MsgBox "Factorielle 0!=1"
   Else
      result = factrec(n)
      MsgBox "Factorielle " & init & "!=" & result
   End If
   Exit Sub
1   resp = MsgBox("Impossible d'exécuter la procédure", vbRetryCancel +
   vbCritical)
   If resp = vbCancel Then
      Exit Sub
   ElseIf resp = vbRetry Then
      GoTo 2 'Attention cela est très dangereux (ne gère pas le conlit des variables: y préférer
le "call")
   End If
End Sub

Function factrec(n As Integer)
   'On utiliSe la méthode récursive
   If n <= 1 Then
      factrec = 1
   Else
      factrec = factrec(n - 1) * n
      debug.print factrec
   End If
End Function

'Avec un boucle "For"
Function factitfor(n)
Dim I As Integer
```

```
   factitfor = 1
   For I = 1 To n
      factitfor = factitfor * I
   Next I

End Function

'----------------------------------------------------------------------------

'ou encore avec un boucle "Do"
Function factitdo(n)
Dim I As Integer
   factitdo = 1
   I = 0
   Do
      I = I + 1
      factitdo = factitdo * I
   Loop While I <> n
End Function

'----------------------------------------------------------------------------

Sub SelectCase()
   Select Case Hour(Time)
      Case 0 To 6
         Message = "Bonne nuit..."
      Case 7
         Message = "Bonjour..."
      Case 8 To 11
         Message = "Bonne matinée..."
      Case 12, 13
         Message = "Bon appétit..."
      Case 14 To 19
         Message = "Bon après-midi..."
      Case Else
         Message = "Bonne soirée..."
   End Select
   MsgBox Message
End Sub

'----------------------------------------------------------------------------

Sub id()
   Dim reponse as String
   reponse = InputBox("Identifiez vous:", "ID Box", "Nom Utilisateur",
   100, 100)
   If reponse = "Maud" Or reponse = "maud" Or reponse = "MAUD" Then
      idok
   ElseIf reponse Like "*soz" Then
      idok
   Else
```

```
        idnul
    End If
End Sub
Sub idok()
    msgbox "C'est ok vous avez été reconnu"
End Sub
Sub idnul()
    msgbox "Access va être fermé", vbcritical
    Quit
End Sub

'------------------------------------------------------------------------

Sub utilisateur()
Dim textlen, renverse, id As String
Dim I As Integer
    id = InputBox("Identifiez-vous")
    'On met en majuscules le UserName
    id = UCase(id)
    'On affiche le tout dans une message box (voi l'aide!! pour le retour
    chariot par exemple)
    'On compte combien de lettres il y a dans le nom de l'utilisateur
    textlen = Len(id)
    'Première structure de boucle de type For
    'Par pas de 1 on analyse en reculant les lettres du nom de l'utilisateur
    For I = textlen To 1 Step -1
        'Vous n'êtes pas obligés de choisir i comme variable d'itération
        'On parcour 1 par 1 les caractères et on les concatène avec le
        caractère précédent
        renverse = renverse & Mid(id, I, 1)
        MsgBox renverse
    Next I
    MsgBox renverse
End Sub

'------------------------------------------------------------------------

'Ce programme renvoie le nombre de voyelles comprises dans un texte
'Objectif: apprendre la command "Mid" + "Like" + "Debug.Print"
'Commandes que l'on retrouve dans les autres logiciels de la suite office
Sub comptevoyelles()
Dim compte As Integer
Dim ch, texte, result As String
    texte = InputBox("Tapez le texte duquel vous voulez enlever les
    voyelles")
    'On initialise une variable (ce qui n'est pas tjrs) obligatoire
    compte = 0
    'On va compter les voyelles
    For I = 1 To Len(texte) 'à comparer avec l'exercice précédent...
        ch = Mid(texte, I, 1)
        'on test la caractère pour voir si c'est une variable
```

```
        If ch Like "[aeiou]" Then
            compte = compte + 1
            'on affiche le résultat intermédiaire dans la fenêtre d'exécution
            Debug.Print ch, I
        Else
            result = result & ch
        End If
    Next I
    'Ecrivez la fonction dans une feuille et appelez dans l'argument une cellule contenant un
texte
    MsgBox "il y avait " & compte & " voyelles"
    MsgBox result
End Sub

'-----------------------------------------------------------------------

Sub afficheascii()
Dim I As Integer
Dim debutascii, finascii As Integer
debutascii = 33
finascii = 126
    For I = debutascii To finascii
        Debug.Print I, Chr(I)
    Next I
End Function

'-----------------------------------------------------------------------

'Ce programme supprime les espaces contenu dans un texte
'Objectif: apprendre à utiliser les valeurs ascii (de 33 à 126)
'Commandes que l'on retrouve dans les autres logiciels de la suite office
Sub suprespaces()
    Dim Temp, ch, texte As String
    texte = InputBox("Tapez une phrase avec des espaces")
    For I = 1 To Len(texte)
        ch = Mid(texte, I, 1)
        '32 est la valeur ascii du l'espace vide
        If ch <> Chr(32) Then
            Temp = Temp & ch
        End If
    Next I
    MsgBox Temp
    'Si vous connaissiez bien les instuctions VB le contenu ci-dessus aurait pu s'abréger
    MsgBox Replace(texte, " ", "")
End Sub

'-----------------------------------------------------------------------

Sub divisedeux()
'Essayez après en changeant "double" en "integer"
Dim nb1, nb2, resultat As Double
```

```
On Error GoTo GestionErreurs
   nb1 = InputBox("Saisissez le dividende")
   nb2 = InputBox("Saisissez le diviseur")
   If IsNumeric(nb1) = False Or IsNumeric(nb2) = False Then
      MsgBox "Une des deux entrées n'est pas un nombre", vbCritical + vbRetryCancel,
"Attention!"
   End If
   resultat = nb1 / nb2
   MsgBox "Le résultat de la division est " & resultat
   'N'oubliez pas de quitter la fonction sinon quoi la gestion des erreurs va être executée
   Exit Sub
GestionErreurs:
   MsgBox Str(Err.Number) & ": " & Err.Description, , "Erreur"
   'Testez la fonction avec une division par zéro, et des saisies de caractères
End Sub

'-------------------------------------------------------------------------

Public Function HMStoSec(strHMS As String) As Long
   HMStoSec = Split(strHMS, ":")(0) * 3600 + _
            Split(strHMS, ":")(1) * 60 + _
            Split(strHMS, ":")(2)
End Function
```

Voyons maintenant une série de codes propres à MS Access que l'on me demande très fréquemment.

19.8 Appliquer le filtre par formulaire automatiquement

Un certain nombre de clients créent des formulaires pour le filtrage uniquement. Or, ces mêmes formulaires sont souvent connectés directement sur les tables. Ils souhaiteraient dès lors que dès qu'un employé commence à écrire dans un champ (alors qu'il ne devrait pas!), sa saisie soit annulée et que le filtre par formulaire soit activé.

Pour faire cela il suffit sur chaque champ associer l'événement suivant dans le code VBA du formulaire:

```
Private Sub Nom_Champ_KeyDown(KeyCode As Integer, Shift As Integer)

  RunCommand acCmdFilterByForm

End Sub
```

19.9 Appliquer un filtre avec critères sur un formulaire

C'est une technique puissant mais qui nécessite dans la réalité un code relativement long pour gérer les combinaisons des champs et le choix d'opérateurs par les employés.

L'idée est que les employés peuvent saisir des critères dans des champs et un clic sur un bouton va filtrer les données du formulaire (qu'il soit tabulaire ou en feuille).

La commande VBA utilise la partie WHERE du SQL (on peut donc y mettre des LIKE, AND ou OR) il faut donc être à l'aise avec ce langage aussi!

DoCmd.ApplyFilter "", "[Nom du champ à filtrer]='" & Me.NomChampAvecCritere & "'"

ou par exemple:

DoCmd.ApplyFilter "", "[Nom du champ à filtrer] LIKE '*' & '" & Me.NomChampAvecCritere & "' '& '*'"

etc.

19.10 Détection formulaire ouvert

```
'Voici une fonction qui s'avère souvent utile dans MS Acccess: comment détecter si un
formulaire est ouvert (pour les constantes voir les tableaux page 407)
Function IsLoaded(ByVal strFormName as String) As Integer
   Const conObjStateClosed=0
   Const conDesignView = 0
   If SysCmd(acSysCmdGetObjectState, acForm, strFormName) <> conObjStateClosed Then
      If Forms(strFormName).CurrentView <> conDesignView Then
         IsLoaded = True
      End If
   End If
End function
'il vous est demandé de faire fonctionner cette function avec un formulaire de votre choix
```

19.11 Détection de la version et compactage

```
'Nous allons détecter ici la version de MS Access sur laquelle nous travaillons, vérifier si la
base est compactée et si ce n'est pas le cas, activer le compactage
Sub Compactage()
   Dim strVersion as String
   Dim blnCompact as Boolean
   strVersion = Application.Version
    msgbox strVersion
   blnCompact = Application.GetOption("Auto Compact")
   If Not blnCompact Then
      Application.SetOption "Auto Compact", True
   End If
'Cherchez maintenant le code nécessaire qui demande à l'utilisateur s'il veut quitter MS
Access et ce de manière esthétique si possible…

End sub
```

19.12 Création d'un bouton ouvrant une page web

```
Private Sub btnPageWeb_Click()
```

```
    Application.FollowHyperlink "http://www.google.fr"

End Sub
```

19.13 Exécution de code SQL (via VBA)

Nous avons vu plus tôt dans le présent support des commandes SQL courantes pour créer, détruire, lier des tables ou des attributs. Mais comment procéder pour faire de même en VBA car c'est très important voir parfois même indispensable de faire des bases de données uniquement avec des requêtes en VBA?

19.13.1 Création de table

La réponse est fort simple et tient en quelques lignes que voici pour une requête de création:

```
Public Sub DoSQL()

   Dim SQL As String
   SQL = "CREATE table tblSuccursale (idSuccursale LONG, strVille CHAR(80),
CONSTRAINT idSuccursalePK PRIMARY KEY (idSuccursale) );"
   DoCmd.RunSQL SQL

End Sub
```

19.13.2 Retour d'une requête

Et pour une requête en VBA pour voir si elle renvoie quelque chose ou rien (contrôle d'existence):

```
Public Sub DoSQL()

   Dim bytRSCount As Integer
   Dim strSQL As String
   Dim dbs As Database

   Set dbs = CurrentDbEnd Sub

   strSQL = "SELECT * FROM tblSuccursale"
   Set rs = dbs.OpenRecordset(strSQL)

  'rsCount retourne le nombre de ligne renvoyée par le requête (si c'est 0 alors évidemment la
requête est vide. On peut aussi faire cela avec un DCount("*","tblSuccursale")
   rsCount = rs.RecordCount

End Sub
```

19.13.3 Boucler sur les enregistrements d'une requête

Et pour lire les enregistrements un par un d'une requête faite en VBA:

```
Public Sub DoSQL()
    Dim rs As DAO.Recordset
```

```
    Set rs = CurrentDb.OpenRecordset("SELECT * FROM tblSuccursale")

    'On contrôle si la requête contient des lignes
    'EOF (End-Of-File) et BOF (Beginning-Of-File)
    If Not (rs.EOF And rs.BOF) Then
        rs.MoveFirst 'une bonne habitude même si pas toujours utile
        Do Until rs.EOF = True
            'On fait par exemple une édition du champ
            rs.Edit
            rs!strVille = "À saisir"
            rs.Update
            'On se déplace au prochaing champ
            rs.MoveNext
        Loop
    Else
        MsgBox "La requête est vide."
    End If

    MsgBox "Fin de la modification de la requête"

    rs.Close 'Fermeture du recordset
    Set rs = Nothing 'Nettoyage

End Sub
```

19.13.4 Source dynamique de formulaire

Et finalement, presque le plus important: faire en sorte que les données qu'affiche un formulaire ou sous-formulaire soit dynamiquement controlé par du VBA:

```
Private Sub Form_Load()

   Dim strNewRecord As String

   strNewRecord = "SELECT * FROM tblSuccursale"
   Me.RecordSource = strNewRecord
   'ou plus généralement
   Forms!MonFormulaire. RecordSource = strNewRecord

End Sub
```

19.13.5 Création d'une table

Voici comment on crée une table tout en VBA:

```
'Création d'une table et définition des champs en utilisant le mot clé New
Sub CreationTable()
    Dim tdfClient As DAO.tabledef
    'Il faut ajouter la référence Microsoft DAO 3.6 Object Library au projet (Outils/Références)
    Dim fl as DAO.Field
    'Crée la définition de la table
```

```
    Set tdfClient = New DAO.tabledef
    tdfClient.Name = "tblEffClients"
    'Crée le 1er champ
    Set fld = New DAO.Field
    With fld
        .Name = "strNomClient"
        .Type = dbText
        .Size = 40
    End With
    tdfClient.Fields.Append fld
    'Crée le 2ème champ
    Set fld = New DAO.Field
    With fld
        .Name = "intEffectif"
        .Type = dbInteger
    End With
    tdfClient.Fields.Append fld
    'Ajoute une clé primaire
    tdfClient.Fields.Append tdfClient.CreateField("ID", dbInteger)
    Set idKey = tdfClient.CreateIndex("IDKey")
    idKey.Primary = True
    idKey.Unique = True
    idKey.Required = True
    idKey.Fields.Append tdfClient.CreateField("ID", dbInteger)
    tdfClient.Indexes.Append idKey
    'Ajoute la table à la base de donnée courant
    Application.CurrentDb.TableDefs.Append tdfClient
    'Réinitialise les variables Objet
    Set tdfClient = Nothing
    Set fld = Nothing
End Sub
```

Sinon une alternative:

```
'À peu près la même chose qu'avant mais en utilisant des méthodes
Sub CreationTable()
    Dim tdfClient As DAO.tabledef
    'Il faut ajouter la référence Microsoft DAO 3.6 Object Library au projet (Outils/Références)
    Dim fl as DAO.Field
    'Crée la définition de la table
    Set tdfClient = CurrentDb.CreateTableDef("tblEffClients")
    'Crée le 1er champ
    With tdfClient
        'Crée le premier champ
        Set fld = .CreateField("strNomClient", dbText, 40)
        .Fields.Append fld
        'Crée le deuxième champ en utilisant une deuxième méthode
        .Fields.Append .CreateField("intEffectif",dbInteger)
    End With
    CurrentDb.TableDefs.Append tdfClient
```

```
    Set fld = Nothing
    Set tdfClient = Nothing
End Sub
```

19.13.5.1 Suppression de toutes les tables

Lors du développement de base Microsoft Access un code utile à mettre dans le back-end de façon subtile, (dans l'événement auto-exec) afin d'éviter un vol trop du back-end et de toutes ses données, consiste à supprimer toutes les relations et tables d'une base de données:

```
Public Function DeleteTables()
    Dim dbs As DAO.Database
    Dim i As Integer

    Set dbs = CurrentDb
    ' Loop backwards through relations
    For i = dbs.Relations.Count - 1 To 0 Step -1
        dbs.Relations.Delete dbs.Relations(i).Name
    Next i
    ' Loop backwards through tabledefs
    For i = dbs.TableDefs.Count - 1 To 0 Step -1
        If Left(dbs.TableDefs(i).Name, 4) <> "MSys" Then
            dbs.TableDefs.Delete dbs.TableDefs(i).Name
        End If
    Next i
    Set dbs = Nothing
End Function
```

19.13.6 Ajout de nouveaux enregistrements

Sinon car trèèèèèèèès important, comment ajouter en programmation un nouvel enregistrement dans un table existante:

```
Private Sub submitButton_Click()

    DoCmd.RunSQL "INSERT INTO tblCustomers (txtCustomerID.Value,
txtCustomerName.Value, txtCustomerAddressLine1.Value, txtCity.Value, txtZip.Value)"

End Sub
```

La syntaxe générale étant:

```
DoCmd.RunSQL "Insert into TaTable (champ1, champ2, champ3) Values ('ValeurTexte',
ValeurNumérique, #ValeurDate#);"
```

19.14 Imports/Exports

19.14.1 Import de données de MS Excel

```
'Vous vous rappelez de la table tblSorties80 que nous devions importer manuellement dans
MS Access ? Si oui, essayez de suite le code suivant:
```

```
Sub btnImport_Click()
   DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel10,
"tblNouveuxVendeurs", _
    "C:\TableNouveauxVendeurs.xls", True, "tblNouveuxVendeurs!A1:D4"
End Sub

'Si des lignes vides persistent vous pouvez faire lors de la requête d'ajout un nettoyage
préalable à l'aide d'une requête de suppression.
```

19.14.2 Export paramétré d'un fichier MS Excel

```
'Il est fréquent que l'on souhaite que l'utilisateur puisse choisir où enregistrer un export.
'Voici un exemple de code:

Sub mcrExportNewsletter()

   On Error Resume Next

   'Ne pas oublier d'ajouter la référence Microsoft Office 11.0 Object Library

   Dim dlgSaveAs As FileDialog
   Dim strFilePath As String

   Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)
   dlgSaveAs.Title = "Select File Location to Export XLSX:"
   dlgSaveAs.InitialFileName = "Data.xlsx"

   dlgSaveAs.Show
   strFilePath = dlgSaveAs.SelectedItems(1)

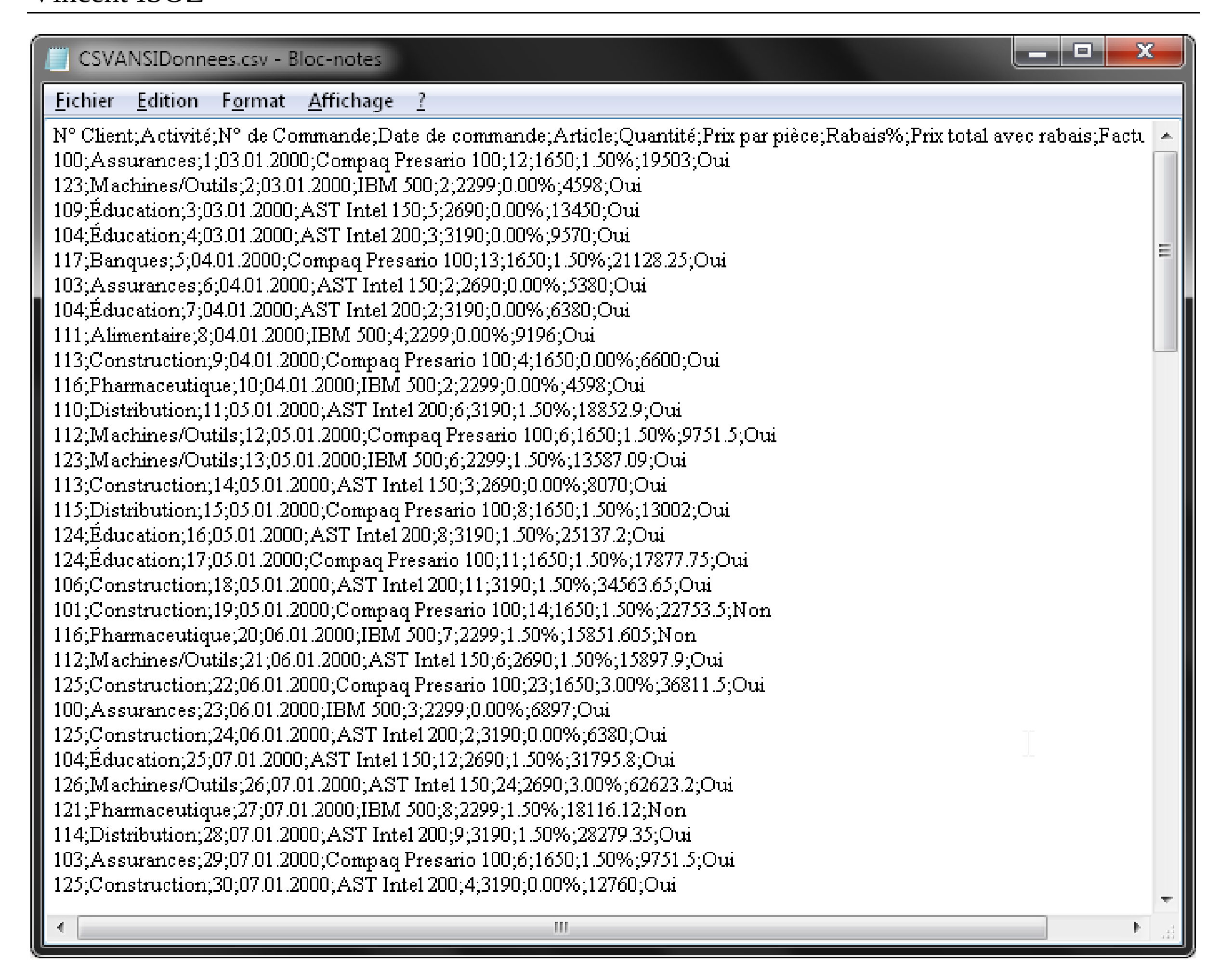
   ' Export de la liste des e-mails de la newsletter
   DoCmd.OutputTo acOutputQuery, "qrData", "ExcelWorkbook(*.xlsx)", strFilePath, True,
"", 0, acExportQualityPrint

End Sub
```

19.14.3 Import d'un fichier CSV

La majorité des fichiers CSV en Suisse sont séparés par des ";" au lieu d'être séparés par des ",". Leur import pose donc problème et nécessite d'enregistrer les étapes de l'assistant manuel d'import de données en tantq que schéma interne au fichier Microsoft Access.

Comme je trouve cette méthode très peu élégante en voici une autre basée sur l'idée que l'on souhaite importer le fichier *.csv suivant:



CSVANSIDonnees.csv - Bloc-notes

Fichier Edition Format Affichage ?

```
N° Client;Activité;N° de Commande;Date de commande;Article;Quantité;Prix par pièce;Rabais%;Prix total avec rabais;Factu
100;Assurances;1;03.01.2000;Compaq Presario 100;12;1650;1.50%;19503;Oui
123;Machines/Outils;2;03.01.2000;IBM 500;2;2299;0.00%;4598;Oui
109;Éducation;3;03.01.2000;AST Intel 150;5;2690;0.00%;13450;Oui
104;Éducation;4;03.01.2000;AST Intel 200;3;3190;0.00%;9570;Oui
117;Banques;5;04.01.2000;Compaq Presario 100;13;1650;1.50%;21128.25;Oui
103;Assurances;6;04.01.2000;AST Intel 150;2;2690;0.00%;5380;Oui
104;Éducation;7;04.01.2000;AST Intel 200;2;3190;0.00%;6380;Oui
111;Alimentaire;8;04.01.2000;IBM 500;4;2299;0.00%;9196;Oui
113;Construction;9;04.01.2000;Compaq Presario 100;4;1650;0.00%;6600;Oui
116;Pharmaceutique;10;04.01.2000;IBM 500;2;2299;0.00%;4598;Oui
110;Distribution;11;05.01.2000;AST Intel 200;6;3190;1.50%;18852.9;Oui
112;Machines/Outils;12;05.01.2000;Compaq Presario 100;6;1650;1.50%;9751.5;Oui
123;Machines/Outils;13;05.01.2000;IBM 500;6;2299;1.50%;13587.09;Oui
113;Construction;14;05.01.2000;AST Intel 150;3;2690;0.00%;8070;Oui
115;Distribution;15;05.01.2000;Compaq Presario 100;8;1650;1.50%;13002;Oui
124;Éducation;16;05.01.2000;AST Intel 200;8;3190;1.50%;25137.2;Oui
124;Éducation;17;05.01.2000;Compaq Presario 100;11;1650;1.50%;17877.75;Oui
106;Construction;18;05.01.2000;AST Intel 200;11;3190;1.50%;34563.65;Oui
101;Construction;19;05.01.2000;Compaq Presario 100;14;1650;1.50%;22753.5;Non
116;Pharmaceutique;20;06.01.2000;IBM 500;7;2299;1.50%;15851.605;Non
112;Machines/Outils;21;06.01.2000;AST Intel 150;6;2690;1.50%;15897.9;Oui
125;Construction;22;06.01.2000;Compaq Presario 100;23;1650;3.00%;36811.5;Oui
100;Assurances;23;06.01.2000;IBM 500;3;2299;0.00%;6897;Oui
125;Construction;24;06.01.2000;AST Intel 200;2;3190;0.00%;6380;Oui
104;Éducation;25;07.01.2000;AST Intel 150;12;2690;1.50%;31795.8;Oui
126;Machines/Outils;26;07.01.2000;AST Intel 150;24;2690;3.00%;62623.2;Oui
121;Pharmaceutique;27;07.01.2000;IBM 500;8;2299;1.50%;18116.12;Non
114;Distribution;28;07.01.2000;AST Intel 200;9;3190;1.50%;28279.35;Oui
103;Assurances;29;07.01.2000;Compaq Presario 100;6;1650;1.50%;9751.5;Oui
125;Construction;30;07.01.2000;AST Intel 200;4;3190;0.00%;12760;Oui
```

dont nous ne connaissons pas obligatoire l'emplacement mais dont le nom est supposé fixe et le nombre de colonnes aussi et fourni avec la ligne d'en-tête de données dans une table existante dans notre base appelée *tblTable* dont les champs sont déjà existants et dont le contenu doit être vidé avant chaque import.

Voici le code qui peut être généralisée relativement facilement:

```
Sub ImportFile()

   On Error Resume Next

   Dim strFilePath As String, strFileName As String, strFullPath As String, strTableName As String, strSeparator As String
   Dim intFieldsNumber As Byte, i As Byte
   Dim valSelect As Variant, MyDB As DAO.Database, MyRS As DAO.Recordset

   'Target table name
   strTableName = "tblTable"
   'Separator for the csv file
   strSeparator = ";"

   Set MyDB = CurrentDb()
   Set MyRS = MyDB.OpenRecordset(strTableName, dbOpenDynaset)
```

```
   'The *.csv file must have the same number of columns as the target table and the columns
must be in the same order!
   'The *.csv file is supposed to have column headers
   strFilePath = ""
   strFileName = "CSVANSIDonnees.csv"

   intFieldsNumber = MyDB.TableDefs(strTableName).Fields.Count
   If strFilePath = "" Then
      strFilePath = GetDirectory
   End If

   strFullPath = strFilePath & "\" & strFileName

   'First we clean the content of the existing target table (main practical situation)
   DoCmd.SetWarnings False
   DoCmd.RunSQL "DELETE * FROM " & strTableName
   DoCmd.SetWarnings True

   Open strFullPath For Input As #1
   row_number = 0
   Do Until EOF(1)
      Line Input #1, LineFromFile
      LineItems = Split(LineFromFile, strSeparator)
      If row_number <> 0 Then
         MyRS.AddNew
         For i = 0 To intFieldsNumber - 1
            MyRS(i) = LineItems(i)
         Next i
         MyRS.Update
      End If
      row_number = row_number + 1
   Loop
End Sub

Public Function GetDirectory(Optional OpenAt As Variant) As Variant
    'Function purpose:  To Browser for a user selected folder.
    'If the "OpenAt" path is provided, open the browser at that directory
    'NOTE:  If invalid, it will open at the Desktop level

   Dim ShellApp As Object

    'Create a file browser window at the default folder
   Set ShellApp = CreateObject("Shell.Application"). _
   BrowseForFolder(0, "Merci de pointer le dossier contenant le fichier de données", 0,
OpenAt)

    'Set the folder to that selected.  (On error in case cancelled)
   On Error Resume Next
   GetDirectory = ShellApp.self.Path
   On Error GoTo 0
```

```
    'Destroy the Shell Application
    Set ShellApp = Nothing

    'Check for invalid or non-entries and send to the Invalid error
    'handler if found
    'Valid selections can begin L: (where L is a letter) or
    '\\ (as in \\servername\sharename.  All others are invalid
    Select Case Mid(GetDirectory, 2, 1)
    Case Is = ":"
        If Left(GetDirectory, 1) = ":" Then GoTo Invalid
    Case Is = "\"
        If Not Left(GetDirectory, 1) = "\" Then GoTo Invalid
    Case Else
        GoTo Invalid
    End Select

    Exit Function

Invalid:
    'If it was determined that the selection was invalid, set to False
    GetDirectory = False

End Function
```

19.15 Automatiser la mise à jour des tables de type BDD liées

Voici un cas classique très utile pour ceux qui déplacent souvent les fichiers de référence des tables liées. D'abord le code qui permet d'afficher une boîte de dialogue et qui demande le fichier où se trouvent dorénavant les tables:

```
Sub cmdFileDialog()
    Dim objDialog As Object
    Dim strPath As String

    On Error GoTo ManageErrors

    Set objDialog = Application.FileDialog(3)

    With objDialog
        .AllowMultiSelect = False
        .Filters.Clear
        .Filters.Add "Access Databases", "*.mdb"
        .Show
        If .SelectedItems.Count = 0 Then
            MsgBox "No file selected."
        Else
            strPath=Dir(.SelectedItems(1))
            Call Attach_Tables(strPath)
        End If
```

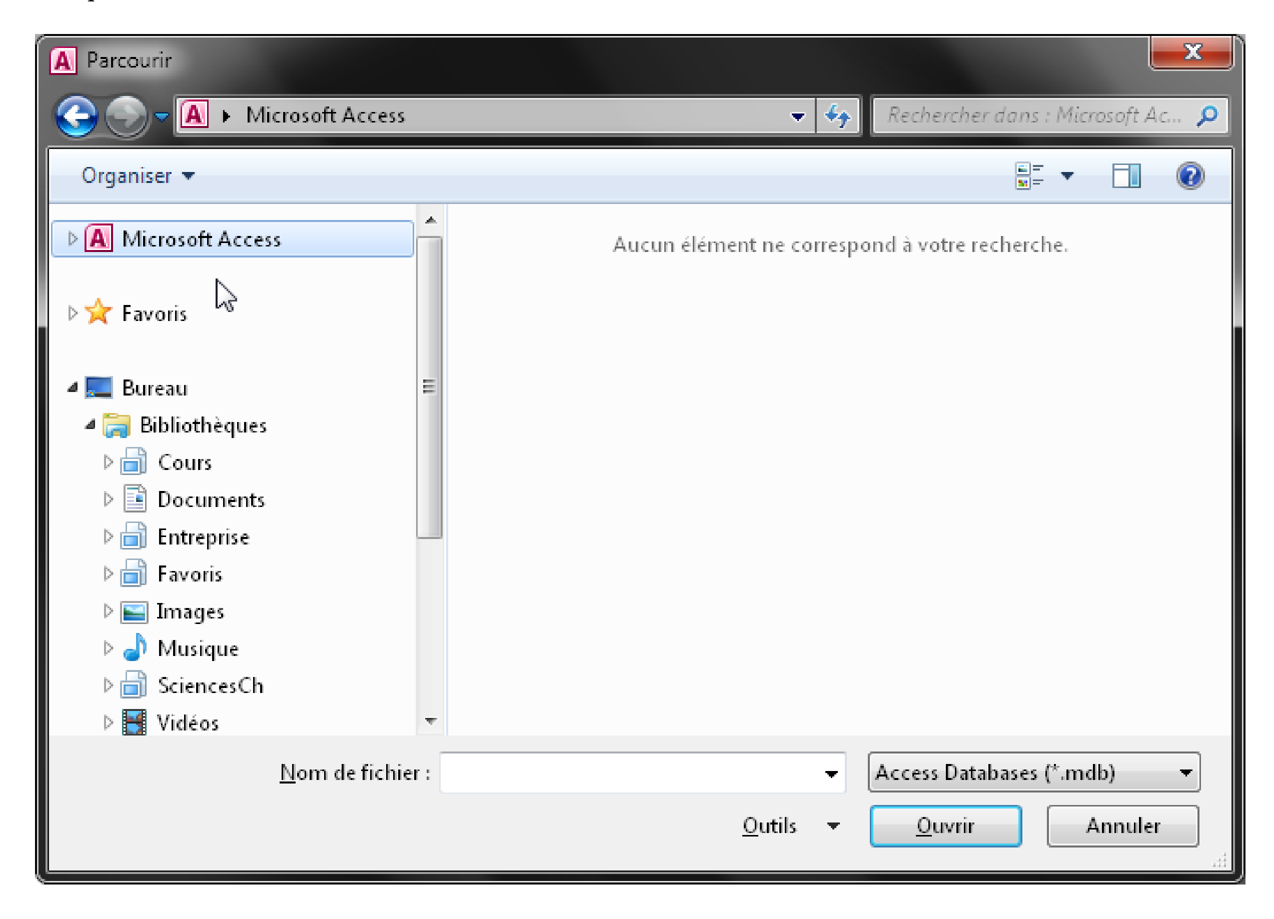
```
    End With
    Set objDialog = Nothing

    Exit Sub

ManageErrors:
        MsgBox "Failure with this version of MS Access! Contact administrator to update the
application", vbCritical + vbOKOnly, "Demo"

End Sub
```

Ce qui donne à l'exécution:



Ensuite, nous récuperons le fichier sélectionné par l'utilisateur et le passons en paramètre dans une autre routine qui va mettre à jour le lien de toutes les tables et qui à l'aide d'une fonction va aussi nous permettre d'afficher si besoin est dans la fenêtre de débuggage les chemins de liaisons précédents:

```
Function GetLinkedDBName(TableName As String)
    'Ne pas oublier la Bibliothèque d'objets Microsoft DAO 3.6
    Dim db As DAO.Database, Ret As Variant, strLength As Integer

    Set db = CurrentDb()
    strLength = VBA.Strings.Len(CStr(db.TableDefs(TableName).Connect))
    If strLength <> 0 Then
        GetLinkedDBName = VBA.Strings.Right(db.TableDefs(TableName).Connect, strLength
- 10)
```

```
        Debug.Print GetLinkedDBName
    End If
End Function

Sub Attach_Tables(strPath As Variant)

    Dim db As Database
    Dim TD As TableDef
    Dim i As Integer, Reponse As Variant, Ret As Variant
    Set db = DBEngine.Workspaces(0).Databases(0)

    Reponse = SysCmd(acSysCmdInitMeter, "Attache les tables", db.TableDefs.Count - 1)

    'boucle sur toutes les tables, réattachant celles dont chaîne
    'de connection est non vide, les autres étant des tables locales

    For i = 0 To db.TableDefs.Count - 1
        Set TD = db.TableDefs(i)
        GetLinkedDBName (db.TableDefs(i).Name)
        If TD.Connect <> "" Then
            TD.Connect = ";DATABASE=" & strPath
            TD.RefreshLink
            If Err <> 0 Then
                MsgBox TD.Name
            End If
        End If
        Reponse = SysCmd(acSysCmdUpdateMeter, i + 1)
    Next i

    DoCmd.Hourglass False
    Reponse = SysCmd(acSysCmdClearStatus)

End Sub
```

19.16 Champ de Recherche sur formulaire

Dans MS Access 2007 l'assistant de liste déroulante de recherche a disparu. Heureusement il reste le VBA:

```
Private Sub ListeReference_AfterUpdate()
    Dim rs As Object
    Set rs = NomFormulaireAvecDonneesRecherchees.Recordset
    'Si les valeur cherchée est un chiffre, enlever les apostrophes
    rs.FindFirst "NomChampSurFormulaireRs = '" &
NomFormulaireAvecListeReference.ListeReference & "'"
    If Not rs.EOF Then Me.Bookmark = rs.Bookmark
End Sub
```

19.17 Evénement sur sortie de formulaire (validation des champs de saisie)

Soit à considérer le formulaire *frmFournisseurs* de la table *tblFournisseurs*. Créez comme exercice un code VBA qui à la sortie du champ *Nom Société*:

1. Change la première lettre de chaque mot du champ en une majuscule

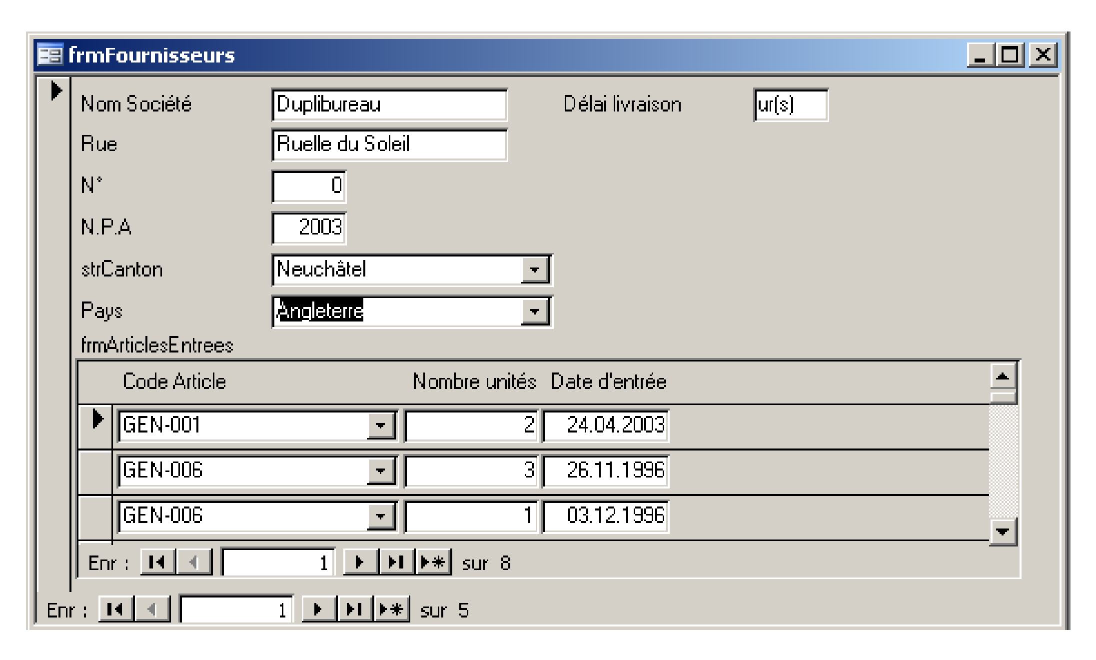
2. Affiche le résultat dans une boîte de dialogue en utilisant les commandes:

```
MsgBox Forms("frmFournisseurs").Controls("strNom").Value
```

ou

```
MsgBox Forms!frmFournisseurs.Controls![strNom].Value
```

En considérant le formulaire *frmFournisseurs* et le sous-formulaire *frmArticlesEntrees* (si vous les avez pas faites-les, l'esthétique ne nous intéresse d'ailleurs pas dans cet exemple):



Nous souhaiterions que tant que le Pays (*strPays*) n'est pas renseigné, les contrôles du sous-formulaire (*tblArticleNb, intNbUnites, datDateIn*) soient bloqués. Pour ceci, sur l'événement *Enter* du champ *tblArticleNb* du sous-formulaire saisissez le code suivant:

```
Private Sub tblArticleNb_Enter()

    If IsNull(Forms!frmFournisseurs!strPays.Value) Then
        Forms!frmFournisseurs!strPays.SetFocus
        Me!tblArticleNb.Locked = True
        Me!intNbUnites.Locked = True
        Me!datDateIn.Locked = True
    Else
        Me!tblArticleNb.Locked = False
        Me!intNbUnites.Locked = False
        Me!datDateIn.Locked = False
    End If

End Sub
```

19.18 Evénement sur fermeture de formulaire (validation des champs de saisie)

En utilisant la commande *Select Case* de VBA et l'instruction *With*, créez une procédure qui lors de la fermeture d'un des formulaires de votre choix par l'action d'un bouton prévu a cet effet contrôle les champs non remplis et:

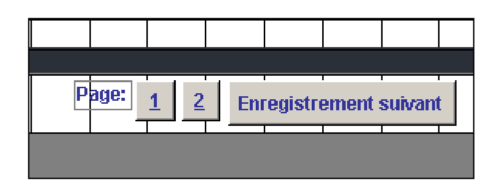
1. Met en rouge les champs non remplis
2. Afficher un message si un des champs n'est pas rempli
3. Change le *Caption* du formulaire en *Correction…*

Aidez-vous de l'exemple suivant:

```
  Dim ctl As Control
  Dim frm As form
  Set frm = Screen.ActiveForm
  For Each ctl In frm
    With ctl
      Select Case.ControlType
      'Etiquettes
      Case acLabel
      .ForeColor = vbBlue
      'Zones de texte
      Case acTextBox
      .ForeColor = vbYellow
      'Listes déroulantes
      Case acListBox, acComboBox
      .ForeColor = vbRed
      End Select
    End With
    frm.Caption = "Ceci est un exemple"
Next ctl
```

19.19 Focus sur formulaire

Créez un bouton sur le formulaire d'ajout de clients de type *Enregistrement suivant* en lui donnant le nom *btnSuivant:*



Remarque: si vous désirez mettre un raccourci sur le "r" de Enregistrement il vous faudra dans les propriétés du bouton dans l'onglet Format et la propriété légende, mettre un & devant la lettre désirée comme on le fait dans Word, Excel, Outlook, etc.

Maintenant effacez ce bouton (il fonctionne évidemment!) et créez un bouton de type *Ajouter enregistrement*

Une fois ce bouton créé avec l'assistant, lorsque l'utilisateur cliquera dessus, le focus sera toujours sur le bouton et malheureusement pas sur le champ *strPrenom*. Ce qui peut s'avérer être une source de perte de temps phénoménale et irritante de plus. Changeons cela! Allez dans le code VBA vous y trouverez:

```
Private Sub newenr_Click()
On Error GoTo Err_newenr_Click


    DoCmd.GoToRecord , , acNewRec
    Prénom.SetFocus
Exit_newenr_Click:
    Exit Sub

Err_newenr_Click:
    MsgBox Err.Description
    Resume Exit_newenr_Click

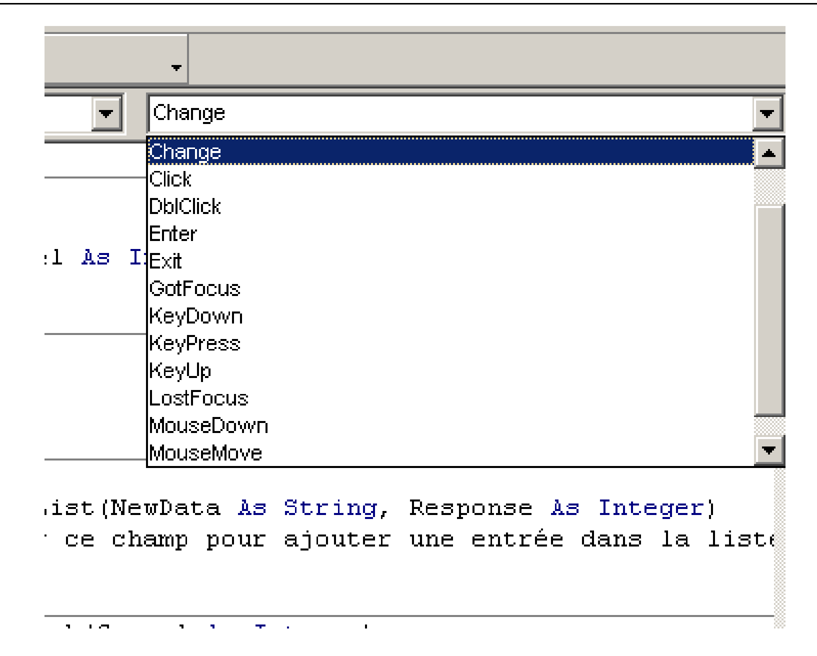
End Sub
```

A la différence que l'on y a déjà ajouté la nouvelle ligne de code (instruction élémentaire) dans le bloc d'instruction a l'endroit adéquat:

strePrenom.SetFocus

Question subsidiare pour les participant: Comment peut on savoir qu'il fallait utiliser SetFocus…??

Maintenant revenons au code de nos boutons et observons les procédures événementielles disponibles. Ces dernières se trouvent toujours au même endroit et l'utilité de certaines d'entre elles peut être contestée mais ne vaut-il pas mieux en avoir trop que pas assez parfois ?

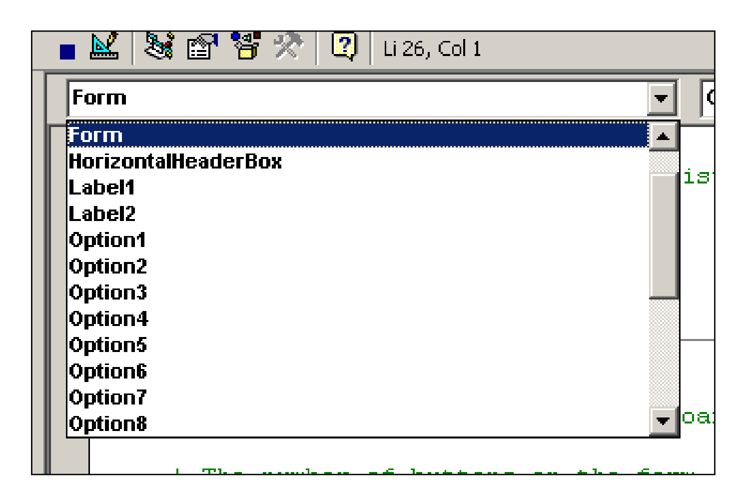


Ces procédures événementielles sont principalement intéressantes lorsque l'on crée un code événementiel sur un nouveau bouton vierge de toute action.

Par exemple, en créant un nouveau bouton avec le code suivant:

```
Sub EditRecord_Click()
    'Active l'édition des enregistrements du formulaire actif
    Me.AllowEdits = True
    Prénom.SetFocus
End Sub
```

Nous avons vu là, un exemple d'événement d'un bouton, mais il en existe également pour le formulaire dans son ensemble. Dès lors à gauche sélectionnez:



Les événements relatifs à "Form" (le formulaire en cours sélectionné dans le projet) sont visibles dans le menu de droite de la feuille de code. Vous comprenez donc que ces événements sont très importants car ils vous ouvrent nombre de possibilités.

Par exemple, nous pourrions ajouter le code suivant (attention!!! prenez l'habitude d'insérer des commentaires dans votre code!!):

```
Sub Form_AfterUpdate()
   Me.AllowEdits = False
   'voir les options de msgbox !!!
   msgbox "L'enregistrement a bien été validé", vbok, "Confirmation"
End Sub
```

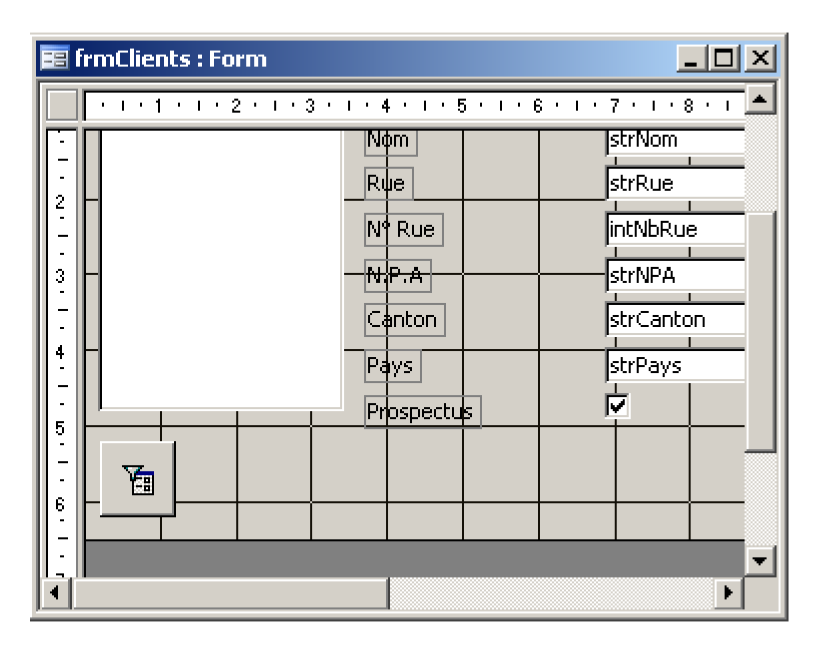
Ce code a pour effet de bloquer l'ajout de nouveaux enregistrements à l'utilisateur une fois que ce dernier en a entré un.

19.20 Filtres

Maintenant regardons ce que nous pouvons faire avec les filtres.

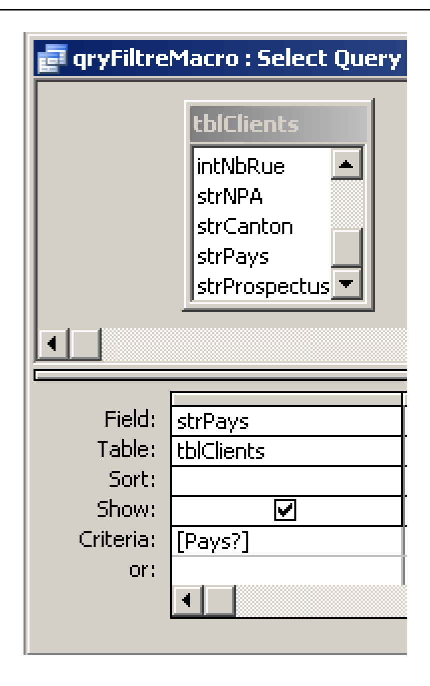
19.20.1 Filtre paramétré par bouton

Créez un joli bouton avec l'image du filtre par formulaire comme ci-dessous sur le formulaire *frmClients* (ce dernier a été crée avec l'assistant de boutons).

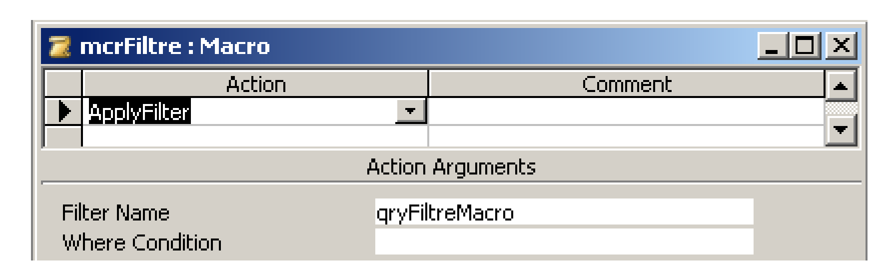


Remarque: Affichez les propriétés de ce bouton et effacez la procédure événementielle à laquelle il est rattaché.

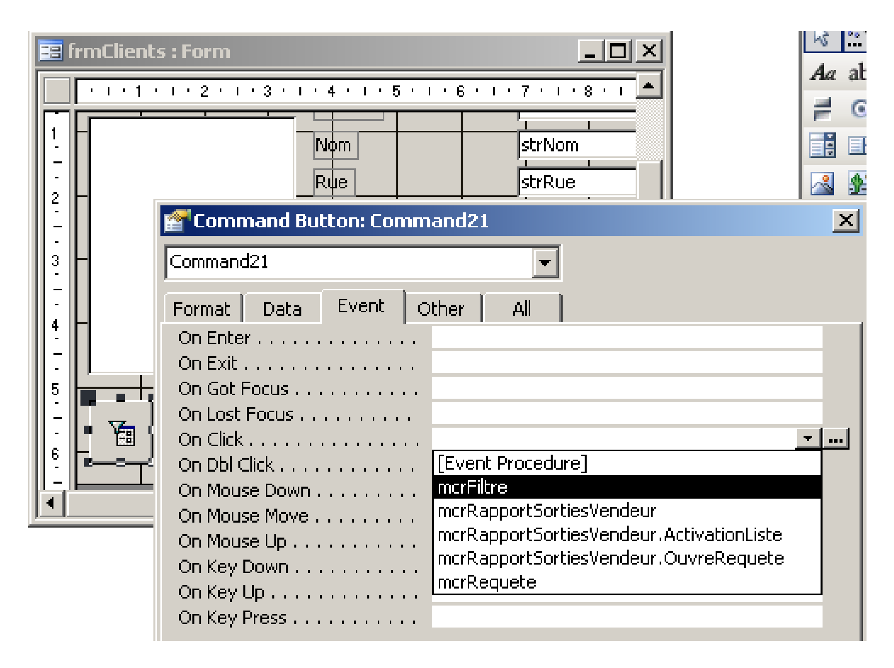
Maintenant créez une nouvelle requête *qryFiltreMacro* utilisant la table *tblClients* utilisant le champ *strPays* comme ci-dessous:



Maintenant nous allons créer une macro avec l'assistant de macro: choisissez *Appliquer un filtre* et informez le nom de la requête qui sera utilisé comme filtre pour le formulaire. Enregistrez ensuite la macro sous le nom *mcrFiltre*:



Revenons maintenant à notre bouton. Affichez les propriétés de ce dernier et dans l'événement *Sur clic* cliquez sur la flèche qui pointe sur le bas et sélectionnez la macro *mcrFiltre*.



Testez maintenant votre bouton (si vous avez tout fait correctement, ce dernier doit parfaitement fonctionner).

Revenons maintenant au VBA.

Affichez les propriétés événementielles du bouton filtre et sur l'événement *Sur clic* cliquez sur les trois petits points… pour y saisir le code suivant:

```
Sub applicfiltr_Click()
   Beep
   MsgBox "Attention, vous allez activer un filtre par pays", _ vbOKOnly + vbInformation,
   "Information"
   DoCmd.RunMacro "mcrFiltre"
End Sub
```

Enregistrez le tout et testez votre bouton.

Maintenant créez comme vous le souhaitez un bouton sur le formulaire qui doit pouvoir désactiver le filtre!

Voici quand même le code:

```
Forms!Contacts.FilterOn = False
```

Vous pouvez travailler en groupe pour trouver la solution. Le premier qui trouve explique aux autres comment il l'a fait. Je validerai la méthode si elle est bonne.

19.20.2 Filtre paramétré par une zone d'options

Maintenant toujours avec les filtres, toujours sur le même formulaire, nous allons créer une zone d'options avec l'outil telle que ci-dessous:



Lors de la création de la zone, n'oubliez simplement pas de donner à l'assistant une valeur à chaque option. Dans cet exemple, nous avons donné la valeur 1 à l'option *Tous les pays* et la valeur *2* à l'option *France*.

Ce groupe d'option ne fait rien si on ne lui ajoute pas du code VBA événementiel. Attachons nous à la tâche. Mais d'abord nommez-le (propriétés de l'ensemble) *filteroptions*

Affichez VBAE à l'écran allez au sommet de la page de code et choisissez dans la liste déroulante des objets *filteroptions* et dans la liste déroulante des événements *AfterUpdate* et saisissez le code suivant:

```
Sub filteroptions_AfterUpdate()
   If filteroptions = 2 Then
```

```
        resp = MsgBox("Vous avez activé le filtre par pays:" & Chr(13) & "France" & Chr(13)
& "Etes vous sûr de vouloir continuer?", vbOKCancel + vbInformation, "Activation Filtre")
        If resp = vbOK Then
            Me.Filter = "[strPays]=France"
            Me.FilterOn = True
        Else
            Exit Sub
        End If
    Else
        Me.FilterOn = ""
        MsgBox "Le filtre est désactivé"
    End If
End Sub
```

19.20.3 Adaptation au filtre par formulaire

Maintenant ce qui fait souci c'est que l'utilisateur active lui-même un filtre dit "par formulaire" (censé être connu). Alors à ce moment, il faudrait que l'on puisse détecter ce que fait l'utilisateur et choisir ce que l'on fait avec nos boutons radios, qui doivent avoir une correspondance logique avec l'activation manuelle du filtre par formulaire.

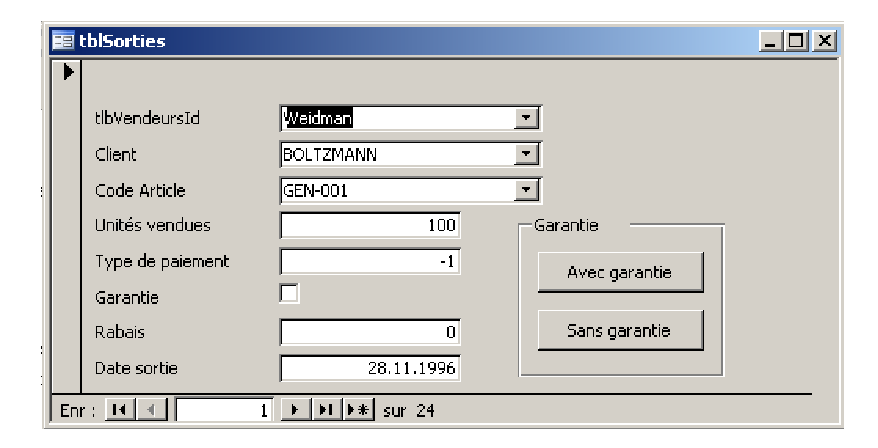
Arbitrairement on choisit de désactiver tous les boutons radios quand le filtre par formulaire sera activé. Pour ce, vous allez dans la page de code du formulaire *frmClients* et tapez le code suivant:

```
Sub Form_ApplyFilter(Cancel As Integer, ApplyType as Integer)
    If ApplyType = acShowAllRecord Then
        'Active le bouton radio "Tous les pays"
         filteroptions = 1
    Else
        'Tous les boutons radios sont désactivés
        filteroptions = Null
    End If
End Sub
```

Nous avons maintenant un système cohérent!

19.20.4 Filtres avec boutons a bascule

Maintenant, avec l'outil *Zones* de la boîte à outils contrôles des formulaires créez le système de boutons à bascule suivant pour le formulaire automatique de la table *tblSorties*:



Lors de la création de la zone, n'oubliez simplement pas de donner à l'assistant une valeur à chaque bouton. Dans cet exemple, nous avons donné la valeur 1 au bouton *Avec garantie* et la valeur *2* au bouton *Sans garantie*.

Nommez cette zone *filteroptions* également et allez dans le VBA du formulaire pour activer l'événement *After_Update* et saisissez:

```
Private Sub filteroptions_AfterUpdate()
   If filteroptions = 2 Then
      Me.Filter = ""
      Me.Filter = "[bolGarantie]=True"
      Me.FilterOn = True
   Else
      Me.Filter = ""
      Me.Filter = "[bolGarantie]=False"
      Me.FilterOn = True
   End If
End Sub
```

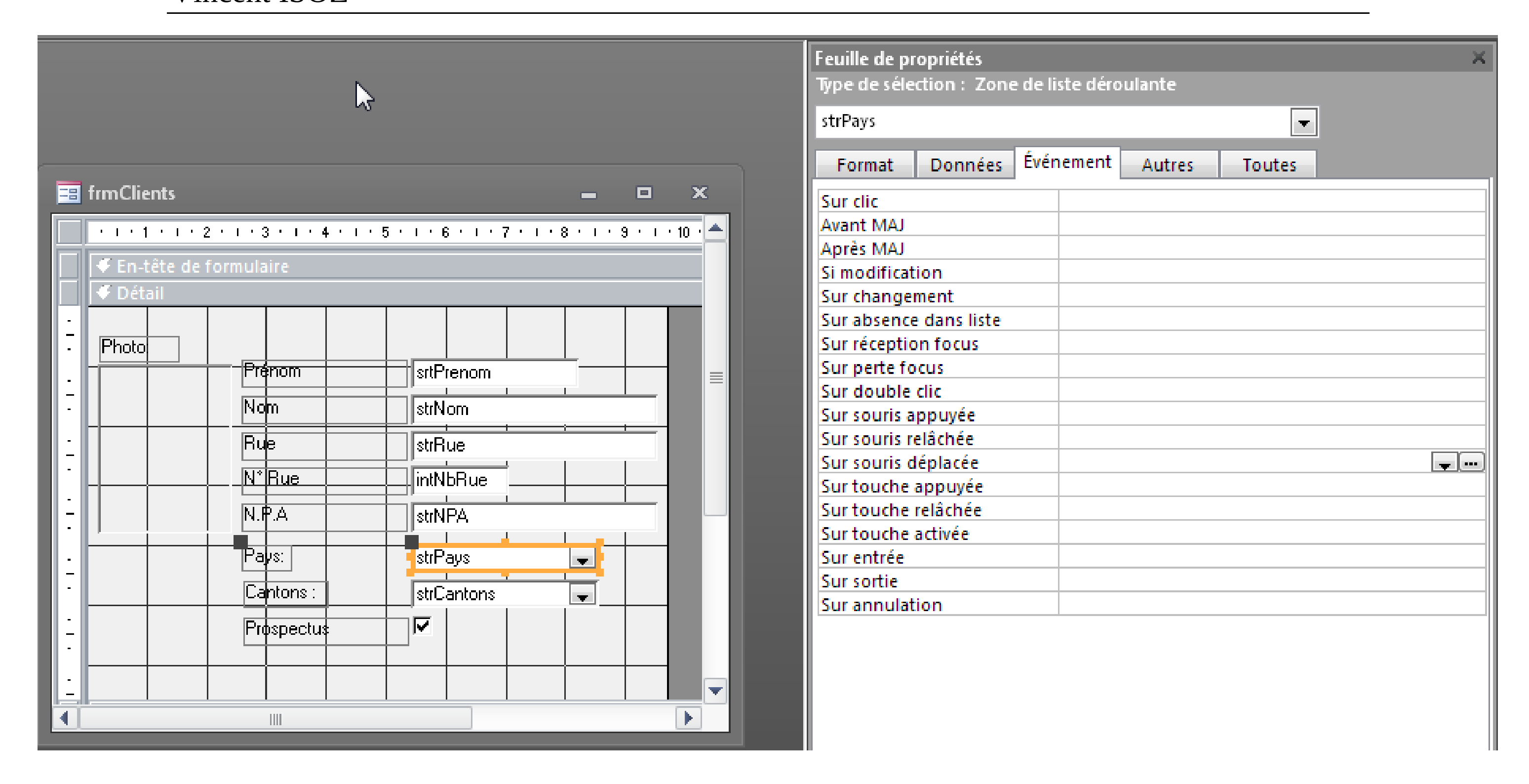
19.21 Listes déroulantes

Nous allons présenter ici quelques codes courants relativement aux listes déroulantes:

19.21.1 Ouvrir une liste déroulante lors du survol avec la souris

Pour faire gagner du temps à vos utilisateurs il peut être pratique d'ouvrir une liste déroulante lors du survol de la souris sur une liste déroulante.

Pour cela, nous allons afficher les propriétés de la liste déroulante qui nous intéresse et cliquons sur la propriété *Sur souris déplacée* (*On mouse move* en anglais):



et nous générons un code VBA:



et nous écrivons:

```
Private Sub strPays_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
   With Me.strPays
      .SetFocus
      .Dropdown
   End With
End Sub
```

19.21.2 Filtrage d'une liste déroulante sur la base d'un champ

Le code de l'exemple ci-après permet d'aficher dans une liste déroulante le nom des clients dont le nom commence par une lettre donnée dans un champ à part.

Pour teste cet exemple vous devez:

1. Créer un formulaire

2. Ajouter les contrôles suivants *fldNom*, *lstNom*, *btnListe1*

```
Private Sub btnListe1_Click()

   Dim strSQL As String

   'Contrôle du nom saisi
   If fldNom = "" Then
      MsgBox "Vous devez saisir au moins une lettre", vbExclamation
      fldNom.SetFocus
   End If

   'Propriétés de la liste déroulante
   Me.lstNom.RowSourceType = "Table/Requête"
   strSQL = "SELECT tblClients.strNom FROM tblClients " & "WHERE (tblClients.strNom
   Like '" & Me.fldNom & "*')"
   Me.lstSoc.RowSource=strSQL

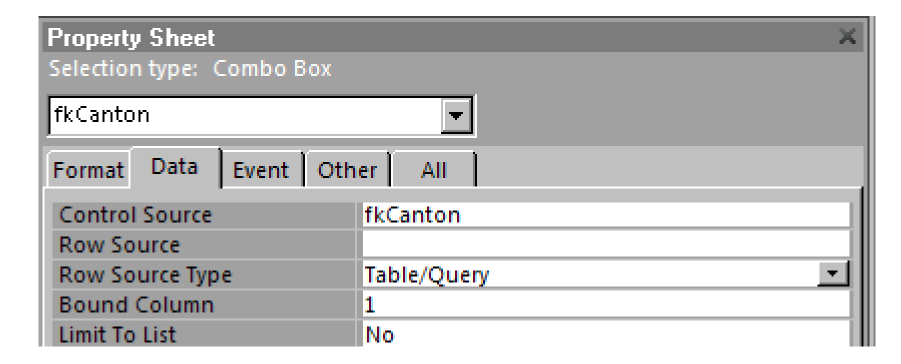
   'Réactualisation des données de la liste
   Me.lstNom.Requery

End Sub
```

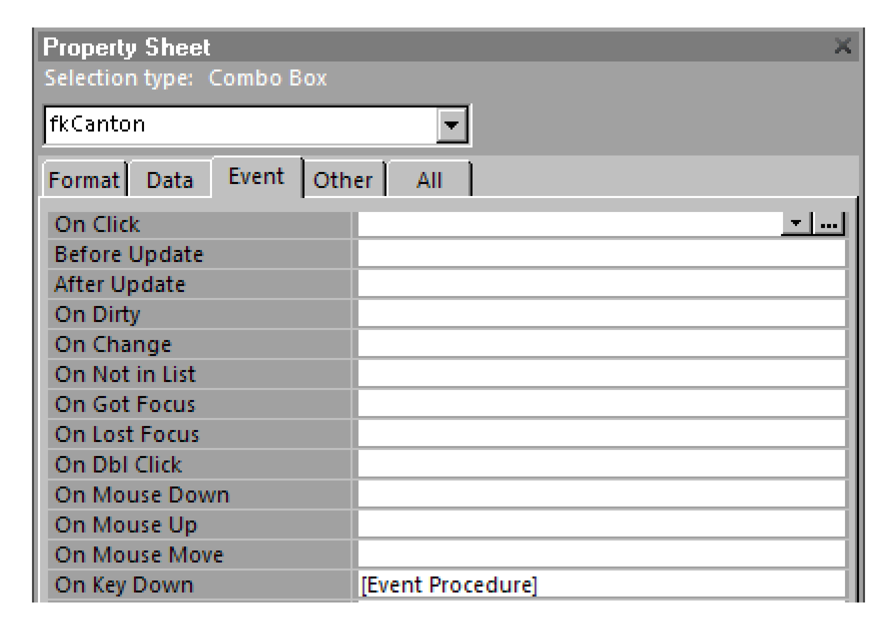
19.21.3 Filtrage d'une liste déroulante sur la de la saisie dans la liste déroulante

L'idée est la même qu'avant mais cette fois-ci nous allons filtrer la liste déroulante des cantons *strCantons* par rapport à ce qui est tapé dans cette même liste car il y a des petites subtilités à prendre en compte pour ce type de scénario très important et très demandé par les créateurs de bases de données dès que le contenu des listes devient non négligeable.

D'abord, il faut aller dans les propriétés de la liste déroulante *strCantons* du formulaire *frmClients* pour mettre sa propriété *Limit to list* (*Limiter à la liste*) à *No* et enlever aussi le code *SQL* se trouvant dans la propriété *Row Source*:



Ensuite, sur l'événement *On Key Down*:



Nous allons donc définir la procédure événementielle suivante en y ajoutant par la même occasion l'événement *Form_Load* à la main:

```
Option Compare Database

Dim strChaine As String

Private Sub fkCanton_KeyDown(KeyCode As Integer, Shift As Integer)
    strChaine = strChaine & Chr(KeyCode)
    'Si l'utilisateur fait un "Home" (code ASCII 36) sur le clavier on considère
    'qu'il veut recommencer et on lui expliquera au préalable que cela fonctionne ainsi
    If KeyCode = 36 Then
        strChaine = ""
        Me.fkCanton = Null
    End If
    sqlCantons = "SELECT tblCantons.strCanton FROM tblCantons WHERE
    tblCantons.strCanton like '" & strChaine & "*'"
    Me.fkCanton.RowSource = sqlCantons
End Sub

Private Sub Form_Load()
    sqlCantons = "SELECT tblCantons.strCanton FROM tblCantons"
```

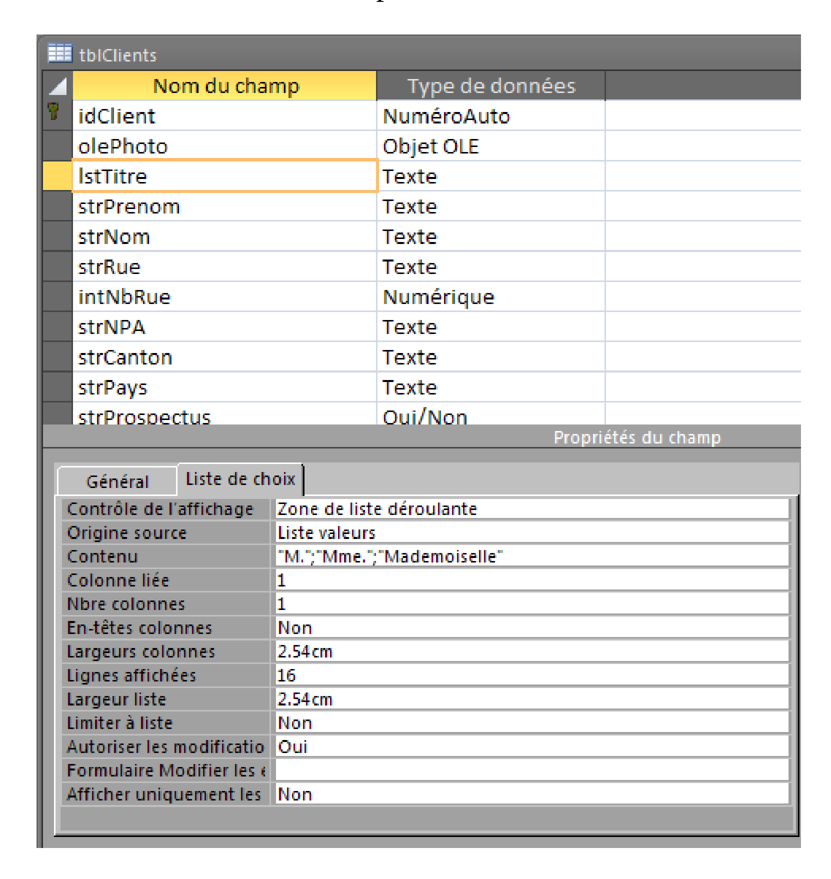
```
    Me.fkCanton.RowSource = sqlCantons
End Sub
```

19.21.4 Ajout directe d'une données dans une liste déroulante

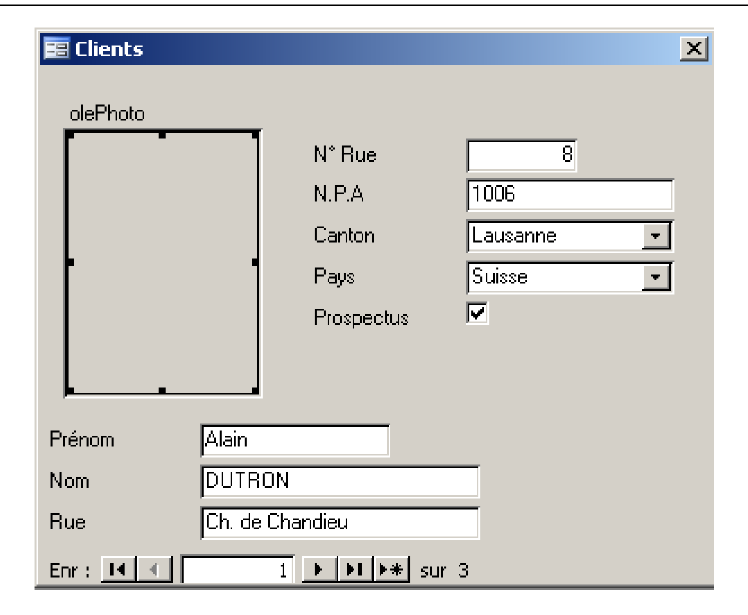
Toujours avec les listes déroulantes… lorsque nous saisissons un nouveau client, il peut manquer un pays dans la liste des pays. Or, comme nous l'avons vu en début de ce support de cours, il est un peu ridicule de créer un formulaire que pour l'insertion de nouveaux pays.

Le mieux dès lors c'est de gérer en VBA l'insertion sur champ de nouveaux éléments dans la table *tblPays*.

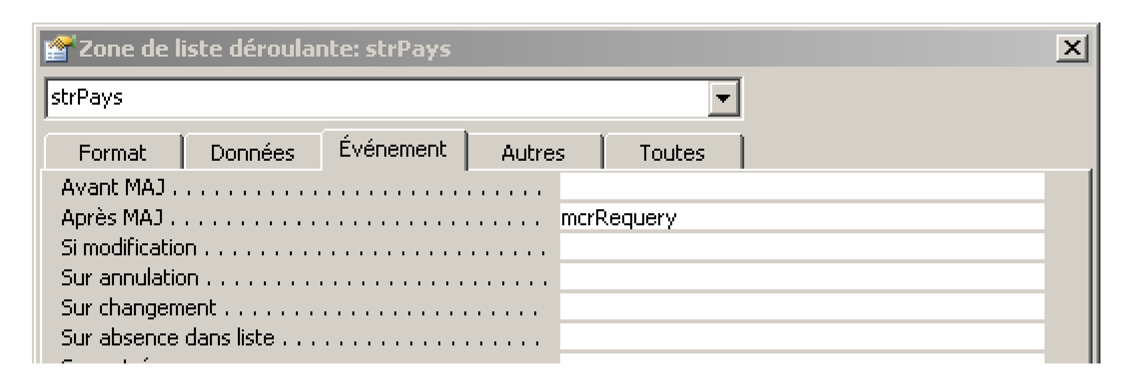
Remarque: Ce code n'est pas utile pour les listes déroulantes de type "statiques" (comprendre: non liéés à des tables conformément à la deuxième forme normale). Effectivement pour ce type de liste il suffit de changer la propriété *Autoriser les modifications* à *Oui* dans les propriétés de la liste de Choix comme nous pouvons le voir ci-dessus:



Considérons donc notre bon, vieux et simple formulaire *frmClients*:



et en particuliers les événements du champ *strPays* (attention!!! prenez garde à ce que le champ *strPays* dans la table *tblClients* ait l'option *Limité à la liste* sur *Oui*):



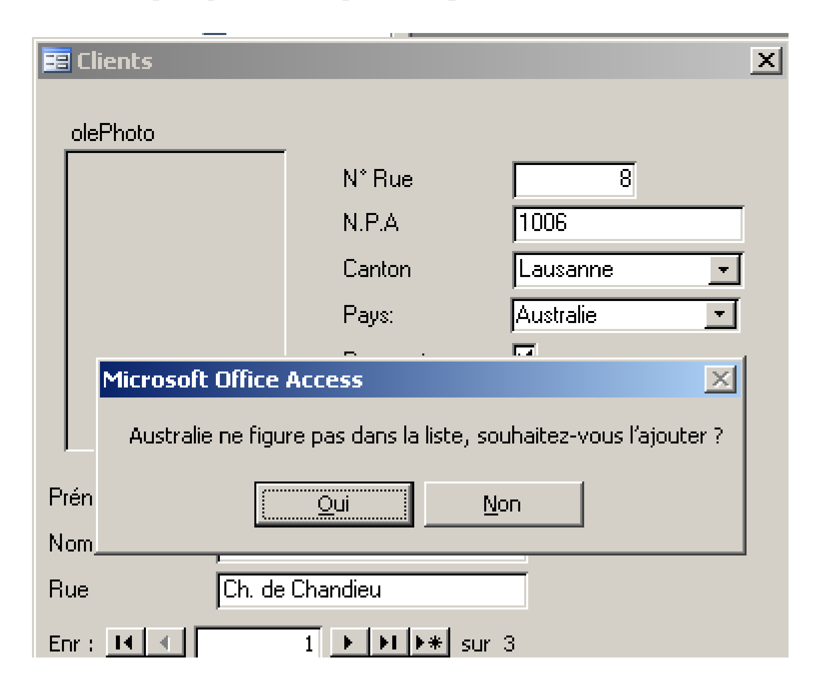
et celui qui se nomme *Sur absence dans liste*. Si vous activez l'édition VBA de cet événement, saisissez-y le code suivant:

```
Private Sub strPays_NotInList(NewData As String, Response As Integer)
   On Error GoTo cboJobTitle_NotInList_Err
   Dim intAnswer As Integer
   Dim strSQL As String
   intAnswer = MsgBox("La région " & Chr(34) & NewData & _
      Chr(34) & " n'existe actuellement pas dans la liste." & vbCrLf & _
      "Souhaiteriez vous l'y ajouter?", vbQuestion + vbYesNo)
   If intAnswer = vbYes Then
      strSQL = "INSERT INTO tblRegions([strRegions]) " & "VALUES ('" & NewData &
"');"
      DoCmd.SetWarnings False
      DoCmd.RunSQL strSQL
      'La commande RunSQL ne fonctionne que pour des requêtes d'action pour info…
      DoCmd.SetWarnings True
      MsgBox "La région a bien été ajoutée à la liste", vbInformation
      Response = acDataErrAdded
   Else
```

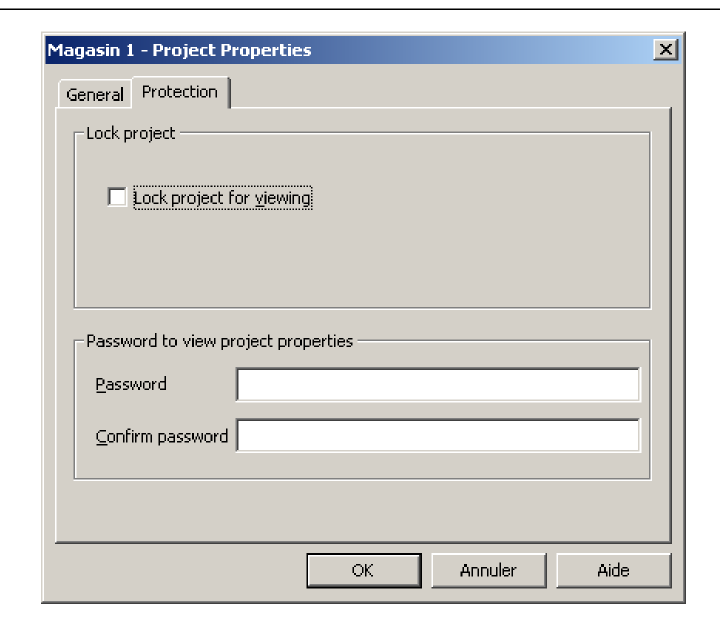
```
      MsgBox "Veuillez choisir un élément existant dans la liste.", vbInformation, Response =
acDataErrContinue
   End If
cboJobTitle_NotInList_Exit:
   Exit Sub
cboJobTitle_NotInList_Err:
   MsgBox Err.Description, vbCritical, "Error"
   Resume cboJobTitle_NotInList_Exit
   intReponse = MsgBox(NewData & " ne figure pas dans la liste, souhaitez-vous l'ajouter ?",
vbYesNo)

End Sub
```

Dès lors, si vous saisissez quelque chose qui n'est pas dans la liste:



Remarque: Il est possible maintenant que vous souhaitiez empêcher tout utilisateur de changer votre code VBA. Pour ce faire, cliquez avec le bouton droit de la souris dans le VBAE sur un des modules et sélectionnez *Propriétés de…*.. Vous y verrez un onglet nommé *Protection*:



Attention n'oubliez pas d'activer la case à cocher et d'y mettre un mot de passe convenable (12 caractères minimum, mélange de majuscule, minuscules et de chiffres).

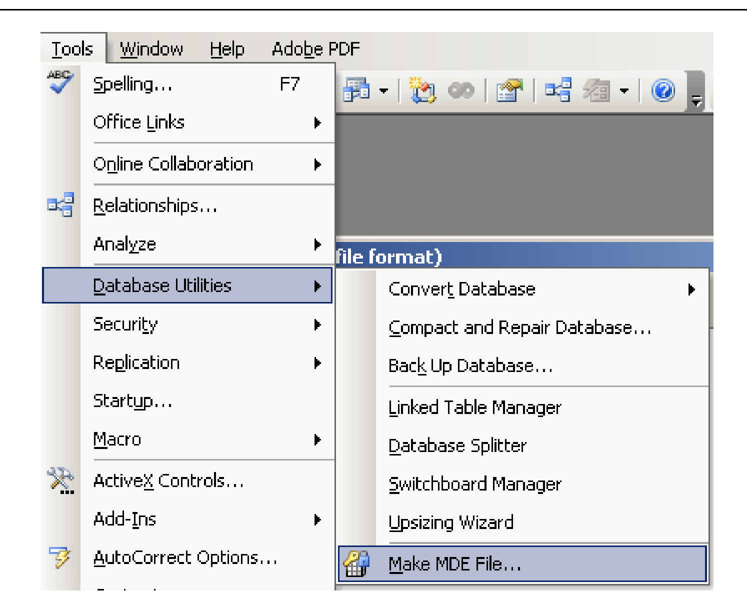
On rappelle également que lorsque l'on utilise du VBA, il est préférable lors du démarrage de l'application MS Access de restreindre au maximum la liberté de l'utilisateur (cela évite les problèmes).

Pour cela (cela est censé être un rappel) allez dans:

Outils/Démarrage…

1. Définissez un titre pour l'application
2. Choisissez une icône (vous pouvez en télécharger sur www.sciences.ch)
3. Choisissez le formulaire à afficher au démarrage de l'application
4. Décochez absolument tout

Au besoin, il est possible de créer un fichier MDE de votre base MS Access (mais faites en une copie de sauvegarde au préalable) ce qui à pour avantage (entre autres) de crypter le code VBA:

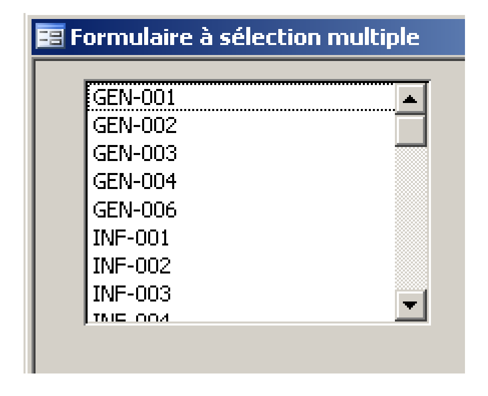


19.21.5 Liste déroulante à choix multiples

Dans cet exemple nous allons voir comment utiliser les listes à choix multiples. L'idée pour faire simple, est de construire un formulaire avec une telle liste qui permet d'exécuter une requête sur plusieurs éléments.

Ainsi, nous allons faire une requête sur la table *tblSorties* qui permet à partir d'un formulaire de choisir le ou les articles que nous souhaitons visualier.

Créez donc un formulaire *frmReqMultiple* avec une liste à choix (dont il faudra dans les popriétés activer la sélection multiple) que nous nommerons *lstArticles* basée sur la liste des articles de la table *tblArticles*.



Créez aussi une requête *qryMultiple* depuis la table *tblSorties*. Vous n'avez pas besoin de spécifier de paramètres ils seront automatiques supprimés avec le code VBA.

Créez enfin un bouton nommé *btnQuery* sur le formulaire *frmReqMultiple* qui ouvre la requête:

Dans l'événement *After_Update* (Après MAJ) de la liste à choix multiple, écrivez le code suivant:

```
Private Sub btnQuery_Click()
```

```
  Dim MyDB As DAO.Database
  Dim qdef As DAO.QueryDef
  Dim i As Integer
  Dim strSQL, strWhere, strIn As String
  Dim varItem As Variant
  Set MyDB = CurrentDb()

  strSQL = "SELECT * FROM tblSorties"

  For i = 0 To lstArticles.ListCount - 1
     If lstArticles.Selected(i) Then
        strIN = strIN & "'" & lstArticles.ItemData(i) & "',"
     End If
   Next i

  strWhere = " WHERE [tblArticleNb] in (" & Left(strIN, Len(strIN) - 1) & ")"
  MyDB.QueryDefs.Delete "qryMultiple"
  Set qdef = MyDB.CreateQueryDef("qryMultiple", strSQL)

  DoCmd.OpenQuery "qryMultiple", acViewNormal

  For Each varItem In Me.lstArticles.ItemsSelected
     Me.lstArticles.Selected(varItem) = False
  Next varItem

End Sub
```

19.22 Autres événements

Nous pouvons à chaque élément de formulaire (ou au formulaire lui-même) définir des événements. Ainsi, si vous affichez les propriétés d'un champ de formulaire ou du formulaire lui-même vous avez comme événements:

Evénements d'un Formulaire
Sur activation
Avant insertion
Après insertion
Avant MAJ
Après MAJ
Si modification
Sur suppression
Avant suppression
Après suppression
Sur ouverture
Sur chargement
Sur redimensionnement
Sur libération
Sur fermeture
Sur activé
Sur désactivé
Sur réception focus
Sur perte focus
Sur clic
Sur double clic
Sur souris appuyée
Sur souris déplacée
Sur souris relâchée
Sur touche appuyée
Sur touche relâchée
Sur touche activée
Aperçu des touches
Sur erreur
Sur filtre
Sur filtre appliqué
Sur minuterie
Intervalle minuterie

Evénements d'un champ de formulaire
Avant MAJ
Après MAJ
Sur changement
Sur entrée
Sur sortie
Sur réception focus
Sur perte focus
Sur clic
Sur double clic
Sur souris appuyée
Sur souris déplacée
Sur souris relâchée
Sur touche appuyée
Sur touche relâchée
Sur touche activée

Evénements d'un Formulaire   Evénements d'un champ de formulaire

Comme vous pouvez le voir, les événements existants sont nombreux et il faudrait plusieurs heures de cours pour faire un exemple intéressant et concret de chaque événement. On vous demandera donc de vous référer à l'abondante littérature disponible dans MS Access.

Définissons un événement *Sur Entrée* de la liste des pays du formulaire *frmClients* de telle façon à ce qu'il redevienne blanc lorsque l'utilisateur ouvre la liste.

Il vous suffit de taper le code suivant à l'endroit où vous savez:

```
Sub strPays_Enter()
   strPays.BackColor=1677215
End Sub
```

Lorsque l'utilisateur sort de la liste déroulante on met cette dernière en gris clair (pourquoi pas?):

```
Sub strPays _Exit()
   strPays.BackColor=12632256
End Sub
```

L'événement qui a lieu lorsque l'utilisateur sort d'un champ de formulaire après avoir changé sa valeur est l'événement: *APRES MAJ* (Mise A Jour)

Voyons maintenant les événements du clavier. Nous allons faire en sorte que lorsque l'utilisateur fait la combinaison de touches CTRL+1 dans les champs *strCanton* et *strPays* se remplissent automatiquement en fonction de notre choix.

Voici le code à saisir dans l'événement du champ ville:

```
Sub strCanton_KeyDown(KeyCode as Integer, Shift as Integer)
   If KeyCode = 49 and Shift=2 then
      strCanton = "Vaud"
      strPays = "Suisse"
   End If

End Sub
```

Evidemment, par choix cette combinaison de touches ne fonctionne que dans le champ *strCanton* et non pas dans le formulaire en entier. Pour ce faire, nous pouvons taper le code suivant:

```
Sub Form_KeyDown(KeyCode as Integer, Shift §)
   If KeyCode = 49 and Shift=2 then
      strCanton = "Vaud"
      strPays = "Suisse"
   End If
End Sub
```

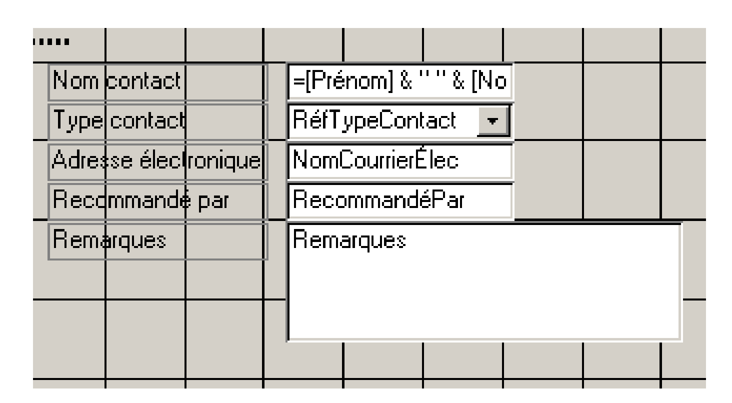
Simple non ?

Quand l'utilisateur essaie de sauver un enregistrement, l'événement *Après MAJ* s'exécute. Ecrivez le code suivant:

```
Sub Form_BeforeUpdate(Cancel as Integer)
   'Si l'utilisateur entre un adresse on fait un check sur le CP
   Dim bytchoice as Byte
   If IsNull(strRue) Xor IsNull(strNPA) Then
      bytchoice = msgbox("Vous avez oublié le NPA. Sauvegarder quand
même?", vbquestion+vbokcancel, "Message")
      If bytchoice = vbcancel Then
         'On pointe le focus sur le NPA
         strNPA.SetFocus
         'On annule l'enregistrement
         Cancel = True
      End If
   End If
End Sub
```

19.23 Fonctions

Maintenant intéressons nous un petit peu aux fonctions et voyons leur utilité. Si vous ouvrez le notre formulaire *frmClients* et passez en mode création. Si vous descendez à l'aide de la barre défilement vous verrez la chose suivante:



On va s'intéresser au code contenu dans le champ *Nom Contact* et le remplacer par un appel de fonction de notre crû.

Basculez en mode VBAE et tapez le code suivant:

```
Function Fullname() As String
   Fullname = Prénom & " " & Nom
End Function
```

Rappel: Le "private" force la fonction à être reconnue uniquement dans ce formulaire. Maintenant en lieu et place du code barbare et peu esthétique du champ de formulaire, tapez-y:

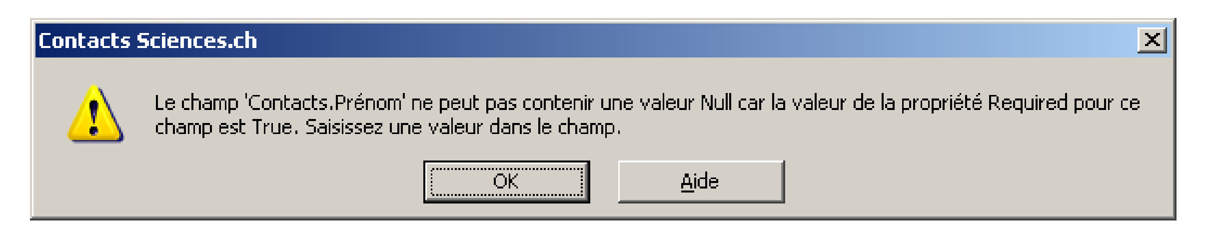
=FullName()

Avouez que c'est plus esthétique quand même!

Si vous souhaitez utiliser une fonction (qui peut faire plusieurs milliers de lignes de code car il n'y a pas de limite à l'imagination) dans plusieurs formulaires il faudra remplacer le "Private" par "Public".

19.24 Gestion des erreurs

Nous avons déjà étudié la gestion des erreurs lors de l'étude de la structure du code VBA (rappelez-vous du code *On Error GoTo GestionErreurs*). Il y a cependant d'autres gestions d'erreurs dans Access. Rappelons le message qui apparaît si notre utilisateur essaie de valider un nouvel enregistrement sans spécifier le nom du client:



Ce qui il faut le dire, est très clair pour un non-initié…

Changeons un peu cela. Passez le formulaire en mode création et affichez les propriétés événementielles du formulaire. Il y a un champ qui se nomme *Sur erreurs* (il est presque en dernier dans la liste). On va y taper du code VBA de façon à avoir la chose suivante:

```
Sub Form_Error(DataErr As Integer, Response As Integer)
   MsgBox "Erreur de saisie! Veuillez contacter votre responsable informatique",
vbRetryCancel + vbCritical, "Message"
   Debug.Print "DataErr= "; DataErr
End Sub
```

Evidemment, nous pouvons faire beaucoup mieux mais à partir de là il suffit d'utiliser son imagination et de la faire correspondre au cahier des charges.

Mais il reste un problème c'est que l'on a un seul et unique message pour toutes les erreurs…? Eh bien, maintenant le code du debug.print va nous permettre d'identifier le numéro de l'exception (ou erreur) engendrée pas Access. Comme vous pouvez le voir il s'agit de l'erreur 3314 qui est l'erreur générée par défaut lorsqu'un champ obligatoire n'a pas été rentré. Dès lors, avec cette méthode on peut faire quelque chose de beaucoup mieux comme:

```
Sub Form_Error(DataErr As Integer, Response As Integer)
   Const ErrFielrequired = 3314
   If DataErr = ErrFielrequired Then
      MsgBox "Erreur de saisie! Veuillez contacter votre responsable informatique",
vbRetryCancel + vbCritical, "Message"
      Response=acDataErrContinue
   Else
      'Force les messages d'erreurs standards d'Access
      Response = acDataErrDisplay
   End If
End Sub
```

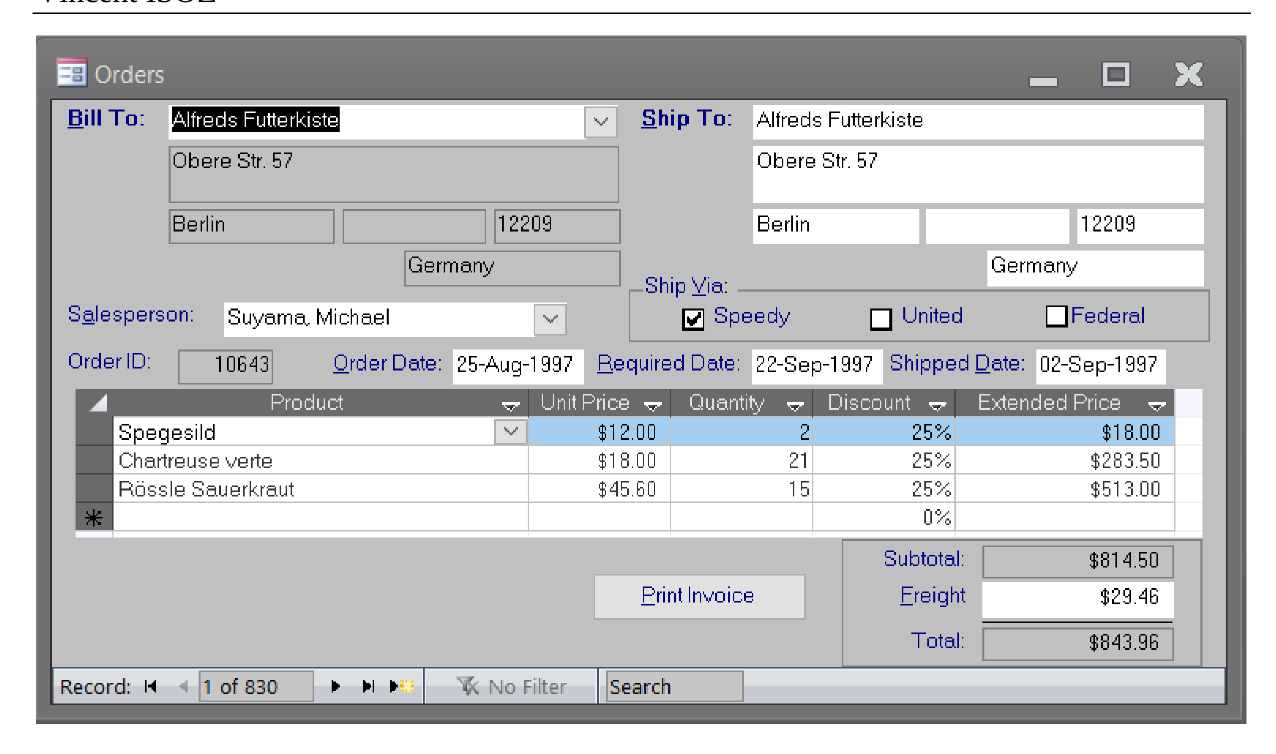
19.25 Duplication de données d'un formulaire avec sous-formulaire

Il arrive parfois que nous ayons besoin de dupliquer les données d'un formulaire avec un sous-formulaire. Or, le bouton par défaut que propose l'assistant de Microsoft Access ne fonctionne qu'avec un formulaire simple.

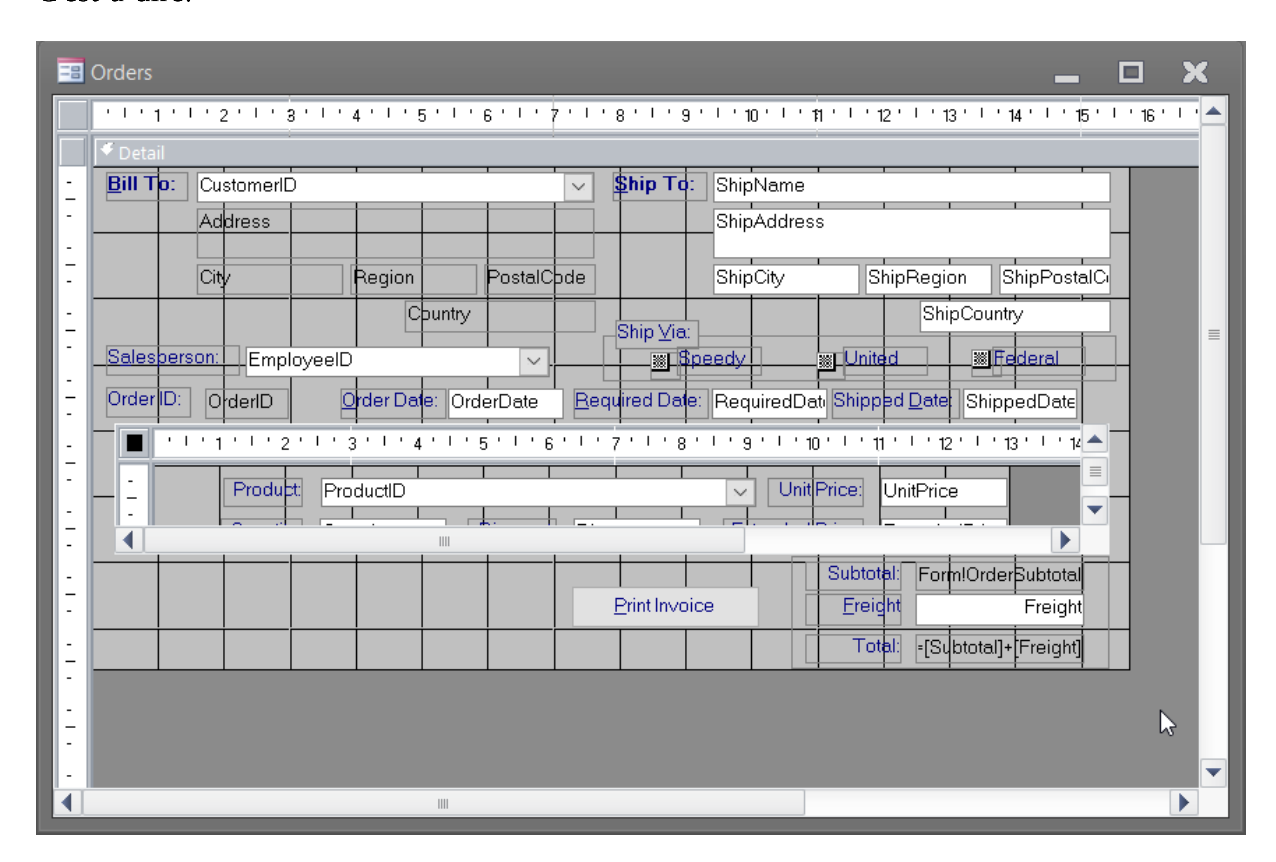
Dès lors, pour voir comment résoudre ce problème voici l'idée[8]:

Considérons le formulaire *Orders* avec sous-formulaire suivant de la base de données Northwind:

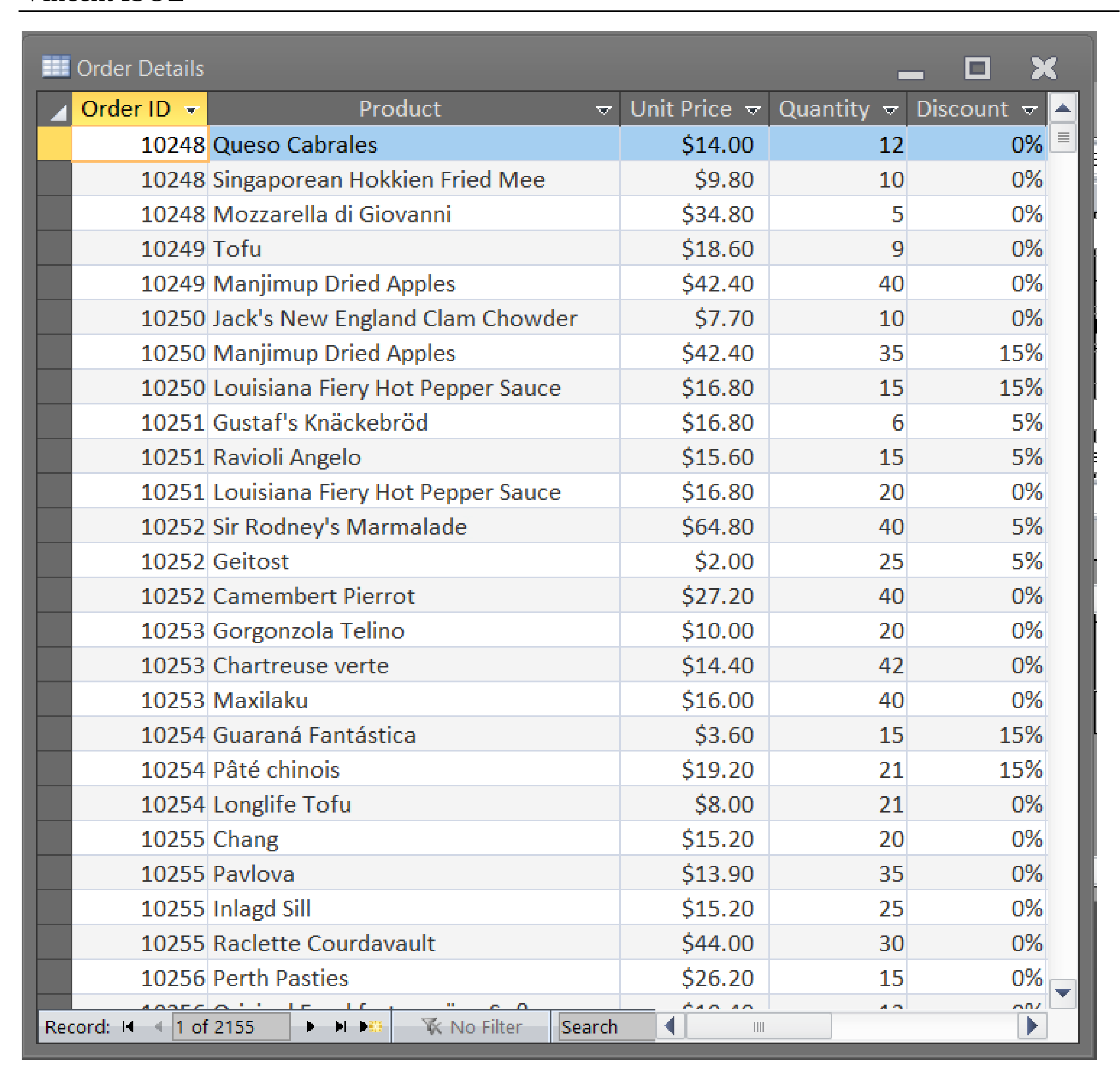
[8] Allen Browne, November 2005, http://allenbrowne.com/ser-57.html



C'est-à-dire:

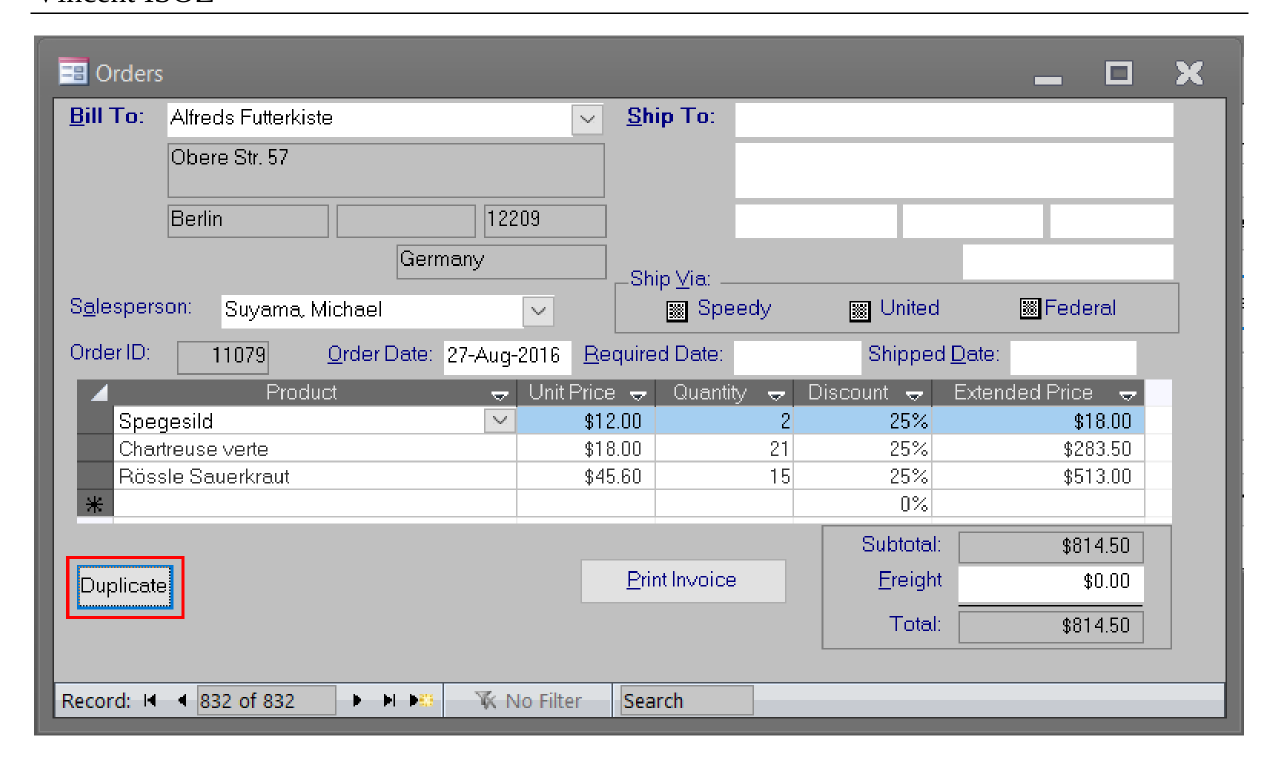


Et la table des commandes détailées y relative:



Order ID	Product	Unit Price	Quantity	Discount
10248	Queso Cabrales	$14.00	12	0%
10248	Singaporean Hokkien Fried Mee	$9.80	10	0%
10248	Mozzarella di Giovanni	$34.80	5	0%
10249	Tofu	$18.60	9	0%
10249	Manjimup Dried Apples	$42.40	40	0%
10250	Jack's New England Clam Chowder	$7.70	10	0%
10250	Manjimup Dried Apples	$42.40	35	15%
10250	Louisiana Fiery Hot Pepper Sauce	$16.80	15	15%
10251	Gustaf's Knäckebröd	$16.80	6	5%
10251	Ravioli Angelo	$15.60	15	5%
10251	Louisiana Fiery Hot Pepper Sauce	$16.80	20	0%
10252	Sir Rodney's Marmalade	$64.80	40	5%
10252	Geitost	$2.00	25	5%
10252	Camembert Pierrot	$27.20	40	0%
10253	Gorgonzola Telino	$10.00	20	0%
10253	Chartreuse verte	$14.40	42	0%
10253	Maxilaku	$16.00	40	0%
10254	Guaraná Fantástica	$3.60	15	15%
10254	Pâté chinois	$19.20	21	15%
10254	Longlife Tofu	$8.00	21	0%
10255	Chang	$15.20	20	0%
10255	Pavlova	$13.90	35	0%
10255	Inlagd Sill	$15.20	25	0%
10255	Raclette Courdavault	$44.00	30	0%
10256	Perth Pasties	$26.20	15	0%

Enfin en ajoutant un nouveau bouton sur le formulaire:



Et en lui associant le code suivant:

```
Private Sub cmdDupe_Click()
    'On Error GoTo Err_Handler
    'Purpose:   Duplicate the main form record and related records in the subform.
    Dim strSql As String    'SQL statement
    Dim lngID As Long       'Primary key value of the new record

    'Save any edits first
    If Me.Dirty Then
        Me.Dirty = False
    End If

    'Make sure there is a record to duplicate.
    If Me.NewRecord Then
        MsgBox "Select the record to duplicate."
    Else
        'Duplicate the main record: add to form's clone.
        With Me.RecordsetClone
            .AddNew
                !CustomerID = Me.CustomerID
                !EmployeeID = Me.EmployeeID
                !OrderDate = Date
                'etc for other fields.
            .Update

            'Save the primary key value, to use as the foreign key for the related records.
            .Bookmark = .LastModified
            lngID = !OrderID
```

```
        'Duplicate the related records: append query.
        If Me.[Orders Subform].Form.RecordsetClone.RecordCount > 0 Then
          strSql = "INSERT INTO [Order Details] ( OrderID, ProductID, Quantity, UnitPrice,
Discount ) " & _
              "SELECT " & lngID & " As NewID, ProductID, Quantity, UnitPrice, Discount "
& _
              "FROM [Order Details] WHERE OrderID = " & Me.OrderID & ";"
          DBEngine(0)(0).Execute strSql, dbFailOnError
        Else
          MsgBox "Main record duplicated, but there were no related records."
        End If

        'Display the new duplicate.
        Me.Bookmark = .LastModified
      End With
   End If

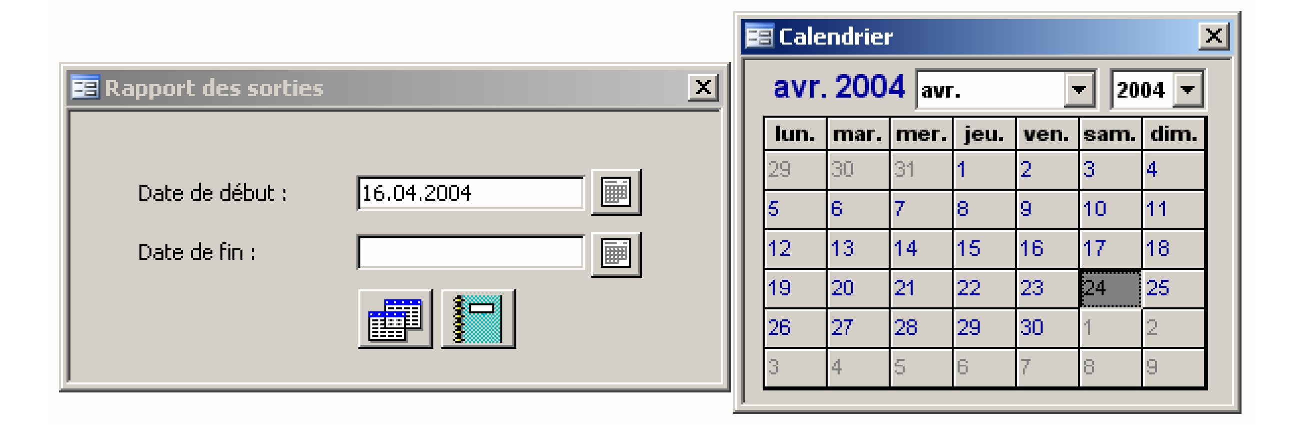
Exit_Handler:
   Exit Sub

Err_Handler:
   MsgBox "Error " & Err.Number & " - " & Err.Description, , "cmdDupe_Click"
   Resume Exit_Handler
End Sub
```

… le lecteur pourra vérifier que tout marche comme attendu!

19.26 Calendriers

Rappelez-vous de la requête que nous avions créée avec le formulaire:



Le système était bon mais avec un formulaire de type calendrier par fomulaire à partir d'un certain moment cela devient indigeste non ?

Avec nos connaissances actuelles nous allons pouvoir considérablement alléger le système afin d'avoir qu'un seul formulaire.

La procédure est la suivante:

Dans un module nommé *modCalendar* écrivez le code suivant:

```
'Varaible globale
Public global_value As Integer

Public Sub RmpFormulaire(insert_value As Integer)
   global_value = insert_value
End Sub
```

Dans le code du bouton d'ouverture du premier calendrier du premier champ rajoutez au début (il faut changer la valeur passée en paramètre à chaque fois entre paranthèses):

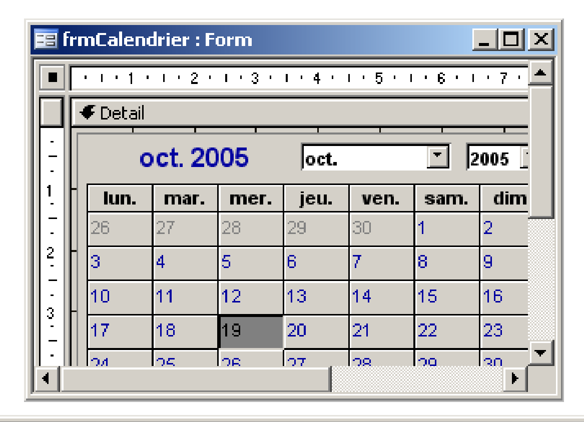
```
Public Sub btnNomBouton_Click()
   modCalendar.RmpFormulaire(1)
End Sub
```

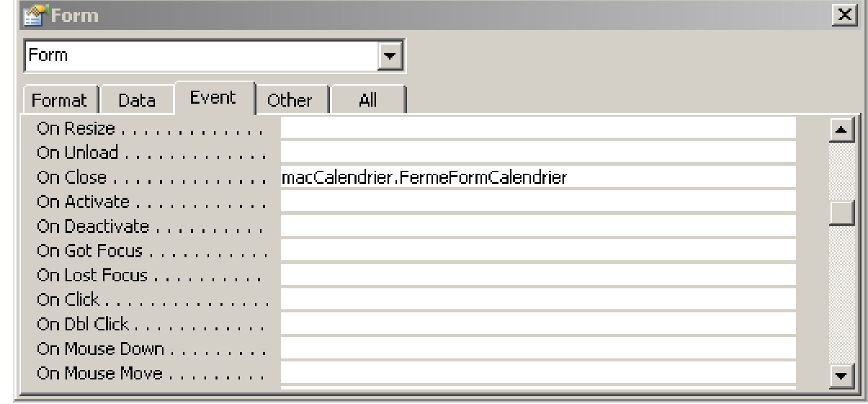
et faites de même avec le deuxième bouton du deuxième champ mais cette fois avec la valeur (2) par exemple. Enfin, dans l'événement OnClose du calendrier saisissez cette fois le code:

```
Private Sub Form_Close()
   If global_value = 1 Then
      Forms!frmNomFormulaireAvecDate!NomChampDateDébut = NomCalendrier.Value
    Else
      Forms!frmNomFormulaireAvecDate!NomChampDateFin = NomCalendrier.Value
    End If
End Sub
```

Pour ceux qui n'aiment pas le VBA, il est toujours possible d'utiliser une petite macro simple (alternative). Effectivement, sur le formulaire *frmRapport* contenant les deux boutons pour ouvrir le calendrier nous avons un champ texte masqué à l'utilisateur et nommé *txtCalendrier*.

Considérons le formulaire *frmCalendrier* avec le calendrier *objCal* ci-dessous avec une macro *macCalendrier* executée à la fermeture du formulaire:





La macro *macCalendrier* contient:

macCalendrier : Macro

Macro Name	Action	
OuvreFormtxtDebutPeriode	OpenForm	Ouvre formulaire calendrier
	SetValue	
	StopMacro	
OuvreFormtxtFinPeriode	OpenForm	
	SetValue	
	StopMacro	
FermeFormCalendrier	SetValue	
	StopMacro	

La procédure est la suivante:

1. Lorsque l'utilisateur clique sur le bouton qui ouvre le formulaire contenant le calendrier, la macro *macCalendrier.OuvreFormtxtDebutPeriode* est executé et fait un *OpenForm* du formulaire contenant le calendrier.

2. Le formulaire avec le calendrier s'ouvre ensuite à l'écran et l'utilisateur clique sur une des dates. Lorsqu'il ferme le calendrier, la macro *FermeFormCalendrier* est executée et l'action *SetValue* effectue l'opération suivante:

   Item: [Forms]![frmRapport]![txtCalendrier]
   Expression: [Forms]![frmCalendrier]![objCal].[Value]

   Cette action a pour effet de prendre la date sélectionné dans le calendrier et de l'écrire dans notre champ masqué du formulaire d'origine.

3. Quand le calendrier est fermé, la macro *macCalendrier.OuvreFormtxtDebutPeriode* continue a s'executer. Elle va ensuite copier la valeur qui se trouve dans notre champ masqué, dans le champ qui correspond au date de début de période tel que:

Item: [Forms]![frmRapport]![txtDebutPeriode]
Expression: [Forms]![frmRapport]![txtCalendrier]

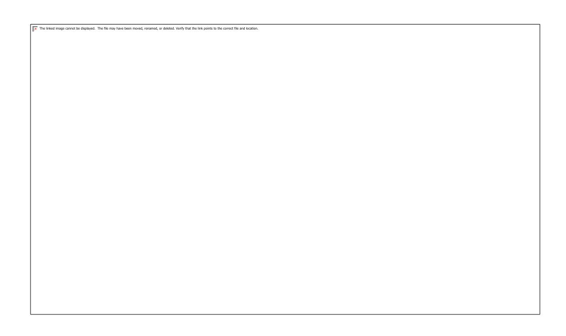
4. Ensuite la macro s'arrête et c'est terminé

19.27 Identification

L'idée est de construire un formulaire d'identification (demandant un mot de passe), pour limiter l'accès à d'autres formulaires ou états. Par exemple:

- L'utilisateur ouvre un formulaire *frmClients* ou un état *rptClients*

- Au préalable, un formulaire *Identification* s'affiche. Tant que l'utilisateur n'a pas fourni un mot de passe correct, il n'accède pas à *frmClients* ou *rptClients*.



L'autre objectif est que ce formulaire soit réutilisable: pas question d'en créer un pour chaque formulaire ou état à "protéger" !

La procédure développée dans les lignes qui suivent doit être vue comme une protection élémentaire. Si vous souhaitez vraiment sécuriser votre base de données, consultez plutôt le chapitre consacré à la sécurité.

Pour des raisons de programmation événementielle, nous simplifierons la gestion du mot de passe en passant par une variable globale, laquelle indiquera si le mot de passe a été trouvé ou non par l'utilisateur:

Dans la fenêtre de Base de données, cliquez sur l'onglet Modules, puis sur le bouton Nouveau.

Tapez le code suivant dans le module:

```
Public blnPasswordOK As Boolean
```

Enregistrez le module sous le nom modPassword, par exemple.

On déclare ici une variable globale (donc visible depuis tout endroit de la base de données). Cette variable est booléenne: elle vaudra False (Faux) tant que le mot de passe n'est pas trouvé, et True (Vrai) si le mot de passe a été tapé correctement.

Construisez maintenant le formulaire d'identification. Le formulaire comporte 3 objets principaux, détaillés ci-dessous.

Il sera enregistré sous le nom frmPassword.

Nom de l'objet	Type d'objet	Icône	Remarques
txtMotDePasse	Zone de texte		C'est dans cette zone que sera tapé le mot de passe. Faites apparaître les propriétés de cet objet et faites en sorte que la propriété **Masque de saisie** soit réglée sur "Mot de passe" (ceci a pour effet d'afficher uniquement des astérisques lors de la saisie).
btnOK	Bouton de commande		Le bouton de validation. C'est ce bouton qui doit vérifier que le mot de passe saisi est correct.
btnAnnuler	Bouton de commande		Un bouton pour fermer le formulaire si on ne connaît pas le mot de passe.

Si le formulaire *frmPassword* s'ouvre, c'est qu'un mot de passe va être demandé puis vérifié. On suppose donc qu'à l'ouverture, le mot de passe n'est pas encore connu.

1. Ouvrez le formulaire *frmPassword* en mode Création.

2. Faites apparaître les propriétés du formulaire, et cliquez sur l'onglet *Evénement*.

3. "Entrez" dans l'événement *Sur ouverture*, et faites en sorte qu'il reprenne le code suivant:

```
Sub Form_Open(Cancel As Integer)
    'Réinitialiser l'état du mot de passe
    ' (blnPasswordOK est une variable globale)
    blnPasswordOK = False

End Sub
```

Le code du bouton *OK*:

1. Faites apparaître les propriétés du bouton btnOK, et cliquez sur l'onglet *Evénement*.

2. "Entrez" dans l'événement *Sur clic*, et tapez ce qui suit:

```
Sub btnOK_Click()
    If IsNull(Me.txtMotDePasse) Then
        MsgBox "Tapez un mot de passe !", vbInformation
        Me.txtMotDePasse.SetFocus
        Exit Sub
    End If
    If Me.txtMotDePasse = "TopSecret" Then
```

```
    ' Fermer la boîte de dialogue "Identification"
        DoCmd.Close
        blnPasswordOK = True
      Else
        MsgBox "Mot de passe incorrect.", vbExclamation
        Me.txtMotDePasse.SetFocus
      End If

End Sub
```

Attention!!! Gérer les utilisateurs et les mots de passe directement dans VBA est un dégueulasse comme méthode. Il vaut mieux avoir une table alouée à ce travail là.

Comme vous le voyez, 2 conditions sont appliquées:

1. La première vérifie qu'un mot de passe a bien été tapé, et ramène sinon sur la zone de saisie.

2. La seconde vérifie que le mot de passe est exact, et ferme le formulaire si c'est le cas. Sinon, un message est affiché, et l'utilisateur est à nouveau ramené sur la zone de saisie.

Comme il a été dit plus haut, il ne s'agit pas d'une sécurité ultime ! Vous notez que le mot de passe est simplement tapé en clair dans le code VBA.

Vous pouvez améliorer le tout en créant une table de mots de passe, et en la lisant par ADO, DAO, SQL ou une fonction de domaine (ce que vous voulez, quoi !), mais dans ce cas, n'oubliez pas que l'utilisateur peut y accéder encore plus facilement.

Le code du bouton Annuler:

Ce code sert juste à fermer le formulaire. Il suffit donc d'écrire:

```
DoCmd.Close
```

Rappelez-vous que la variable blnPasswordOK a été réglée à False lors de l'ouverture du formulaire. Par conséquent, un appui sur le bouton Annuler suppose également que l'utilisateur n'a pas trouvé le mot de passe.

Maintenant que la gestion du mot de passe est en place, il suffit d'y faire appel lorsqu'on ouvre un formulaire ou un état quelconque.

1. Ouvrez un formulaire ou un état.

2. Faites apparaître les propriétés (notamment les événements) de ce formulaire ou de cet état.

3. Programmez l'événement *Sur ouverture* comme suit:

```
Sub Form_Open(Cancel As Integer)
    DoCmd.OpenForm "frm Password", acNormal, , , , acDialog
    Cancel = Not blnPasswordOK
```

```
End Sub
```

Le paramètre acDialog est important:

- acDialog ouvre un formulaire en mode "Boîte de dialogue" (formulaire dit "modal"). Le premier effet est qu'il est impossible de faire quoi que ce soit tant que *frmPassword* n'est pas fermé.

- Le second effet se produit dans le code VB ci-dessus. La ligne Cancel = Not blnPasswordOK ne sera exécutée qu'à la fermeture de ***frmPassword***. De cette façon, on est sûr que le résultat de la variable blnPasswordOK a été défini par *frmPassword*.

Essayez d'enlever acDialog pour voir !

Par ailleurs, c'est la ligne Cancel = Not blnPasswordOK qui annule l'ouverture du formulaire ou de l'état. Cancel est une variable analysée par Access à la sortie de l'événement Form_Open(). Si Cancel = True, l'événement est annulé (ou plutôt, l'ouverture est annulée). Si Cancel = False, les événements s'enchaînent comme d'habitude. Dans notre cas, il faut annuler si le mot de passe n'a pas été trouvé.

Double-cliquez maintenant sur le formulaire ou l'état "sécurisé": la boîte d'identification s'affiche au préalable, et vous devez taper le mot de passe correct. Ceci fonctionne également si le formulaire ou l'état sont ouverts via Visual Basic Application.

Bien sûr, ceci n'empêche pas d'ouvrir vos formulaires en mode Création, ni d'accéder aux tables ou aux requêtes. Pour améliorer le tout, vous pouvez enregistrer votre base au format .mde (menu *Outils / Utilitaires de base de données* ; ceci bloquera l'accès aux formulaires, états et modules ; conservez surtout votre original .MDB !) comme nous l'avons déjà spécifié plus d'une fois.

Encore une fois, tout ceci n'est qu'une protection élémentaire, mais il est vrai que ça peut suffire dans certains cas de figure...

Si nous avions choisi l'option de gérer les mots de passe et utilisateurs via une table, la méthode est tout aussi simple avec juste une petite variante. Il faut d'abord créer une table du *tblUsers* du type suivant:

tblUsers : Table

Nom du champ	Type de données
idUtilisateur	NuméroAuto
strUser	Texte
strPass	Texte

et dans le code VBA de l'événement *btnOK_Click* de changer le vérification du mot de passe par la commande suivante qui va compter s'il l'entrée correspond à au moins un enregistrement de la table *tblUsers*:

```
intNombre = DCount("[strUser]", "tblUsers", "[strUser] = '" & fldLogin.Value & "' AND
[strPass]='" & fldPassword.Value & "'")
```

Après, il suffit d'utilisr *intNombre* comme il convient.

19.28 Utilisation de la sécurité avec VBA

Voici un inventaire non-exhaustif de quelques procédures VBA très utiles liées à la sécurité:

```
'**********************
'Changement du mot de passe de la base de données courante
'**********************

Sub ChangementPass()

CurrentDb.NewPassword "Ancien_pass", "Nouveau_pass"

End Sub

'**********************
'Changement du mot de passe Changement du mot de passe de l'utilisateur courant
'**********************

Sub ChangementPassUser()

With DBEngine.Workspaces(0)
   .Users(.UserName).NewPassword "ancienmotdepasse", "nouveaumotdepasse"
End With

End Sub

'**********************
'Récuperer le nom de l'utilisant courant dans Access
'**********************

Sub RecupUserAccess()

   MsgBox Application.CurrentUser

End Sub

'**********************
'Récuperer le nom de l'utilisant courant dans Windows
'**********************

Sub RecupUserWindows()

   MsgBox Environ("USERNAME")

End Sub

'**********************
'Créer un groupe utilisateur
'**********************

Sub CreerGroupe()
```

```
    'nécessite librairie DAO
    Dim wrk As Workspace
    Dim grp As DAO.Group
    Set wrk = DBEngine.Workspaces(0)

    With wrk
        Set grp = .CreateGroup("Mongroupe", "AZERty12456")
        Groups.Append grp
    End With

End Sub

'***********************
'Créer un utilisateur
'***********************

Sub CreerUtilisateur()

    'nécessite librairie DAO
    Dim wrk As Workspace
    Dim usr As DAO.User
    Set wrk = DBEngine.Workspaces(0)

    With wrk
        Set usr = .CreateUser("Jacques", "567AzeRTY89", "mon_motdepasse")
        .Users.Append usr
    End With

End Sub

'***********************
'Supprimer un groupe utilisateur
'***********************

Sub SupprGroupe()

    'nécessite librairie DAO
    Dim wrk As Workspace
    Set wrk = DBEngine.Workspaces(0)

    With wrk
        .Groups.Delete "Mongroupe"
    End With

End Sub

'***********************
'Supprimer un utilisateur
'***********************

Sub SupprUtilisateur()
```

```
    'nécessite librairie DAO
    Dim wrk As Workspace
    Set wrk = DBEngine.Workspaces(0)

    With wrk
       .Users.Delete "Jacques"
    End With

End Sub

'***********************
'Affecter un utilisateur à un groupe
'***********************

Sub AffectUserGroupe()

    'nécessite librairie DAO
    Dim Wrk As Workspace
    Dim Usr As DAO.User
    Dim Grp As DAO.Group
    Dim grpAffect As DAO.Group
    Set Wrk = DBEngine.Workspaces(0)

    With Wrk
        Set Usr = .Users("Jacques")
        Set Grp = .Groups("Mongroupe")
        Set grpAffect = Usr.CreateGroup("Mongroupe")
        Usr.Groups.Append grpAffect
    End With

End Sub

'***********************
'Lister des groupes et des utilsateurs
'***********************

Sub ListeGrpUsr()

   Dim Wrk As Workspace
   Dim Grp As DAO.Group
   Dim Usr As DAO.User
   Set Wrk = DBEngine.Workspaces(0)

    With Wrk
    Debug.Print "Groupes:"
        For Each Grp In .Groups
            Debug.Print "  " & Grp.Name
            Debug.Print "   Contient les membres suivants:"
            If Grp.Users.Count <> 0 Then
                For Each Usr In Grp.Users
                    Debug.Print "    " & Usr.Name
                Next Usr
```

```
            Else
                Debug.Print "    Aucun Membre"
            End If
        Next Grp
    End With

End Sub

'**********************
'Bloquer l'utilisation de la touche Shift au démarrage (code à activer qu'une seule fois)
'Pour réactiver la fonctionnalité, il faudra créer un bouton qui sera visible que par certains
utilisateurs…
'**********************

  Sub DisableShiftKey()

    Dim db As DAO.Database
    Dim prp As DAO.Property

    Set db = CurrentDb
    db.Properties.Delete "AllowBypassKey"
    Set prp = db.CreateProperty("AllowBypassKey", dbBoolean, False, True)
    db.Properties.Append prp

    db.Properties.Refresh
    Set prp = Nothing
    Set db = Nothing

  End Sub
```

19.29 Automation

19.29.1 Ouvrir MS Excel et y faire des traîtements

```
Sub OuvreFichierExcel1()

  Dim strPath, strFileName As string
  Dim objExpWorkb As Workbook

  strPath="C:\"
  strFileName="FichierExcelOuEcrire.xls"
  Set objExpWorkb = Excel.Workbooks.Open(strPath & strFileName)

  objExpWorkb.Worksheets("sheet1").Range("A1").Value="ça marche ! O_o"

  objExpWorkb.Save
  objExpWorkb.Close
   Set objExpWorkb = Nothing

'Et on ouvre le fichier...

  Call Shell("C:\Program Files\Microsoft Office\OFFICE11\excel.exe c:\test.xls", 1)
```

```
End Sub
```

19.30 D.A.O. et ADO (avec MS Word ou MS Excel)

19.30.1 D.A.O

Les objets d'accès aux données sont très utiles: ils permettent via VBA de manipuler les données (recherches, mises à jour, ajouts d'enregistrement…) de base de données locales ou distantes.

Dans les versions antérieures à MS Access 2000, le seul modèle d'accès aux données était le modèle D.A.O.: Data Access Objects

Avec MS Access 2000 est apparu le modèle A.D.O.: ActiveX Data Objects (pour plus de détails sur les technologies OLE se référer au cours VBA MS Excel du même auteur).

Microsoft recommande d'utiliser le modèle ADO pour les raisons suivantes:

- Meilleure prise en charge de SQL Server
- Meilleures performances en environnement client/serveur
- Code VBA plus court et plus simple

Si vous développez une nouvelle application, vous pouvez utiliser le modèle ADO même si votre application accède à une base de données locale de type MS Access. Si vous souhaitez par la suite migrer vos données sous MSDE (Microsoft Data Engine) ou SQL Server, votre code fonctionnera de façon optimale.

La bibliothèque d'objets D.A.O. prend en charge deux environnements de bases de données différentes, appelées "espaces de travail":

1. Les espace de travail "Microsoft Jet": ils permettent d'accéder à des bases de données de type MS Access, à des serveurs de bases de données ODBC (Open Database Connectivity) et à des bases de données externes telles que dBase, MS Excel et Paradox accessibles via un pilote ISAM (Indexed Sequential Access Method)

2. Les espaces de travail ODBCDirect: ils permettent d'accéder à des serveurs de base de données ODBC de façon "directe", c'est-à-dire sans charger le moteur de base de données Microsoft Jet. Cet espace de travail est par conséquent recommandé pour exécuter des requêtes, des procédures stockées, ou des fonctions spécifiques à ODBC sur un serveur distant de type SQL Server.

Fréquemment en VBA, on peut avoir besoin de lire ou de parcourir le contenu d'une table afin d'y extraire ou d'y écrire une donnée et de manipuler cette dernière afin de l'afficher à l'écran ou de la remplacer par une autre.

Pour cela, le code devient un petit peu plus complexe. Prenons le morceau de code suivant:

19.30.1.1 Méthodes de lecture et écriture de tables et requêtes en DAO

```
Sub LireTable()
```

```
    'N'oubliez pas d'ajouter si ce code ne fonctionne pas la référence Microsoft D.A.O 3.6 (et
non pas la Access Object Library!)


    'Variable qui contient ce que l'on veut insérer dans la table (string arbitrairement)
    Dim NewData As String
    'le dbs est ici pour signifier DataBaSe (cela simplifie la relecture)
    Dim dbs As DataBase
    'le rst est ici pour signifier RecordSeT (cela simplifie la relecture)
    Dim rstNomLigne As DAO.Recordset
    Dim tdfNomTable As TableDef
    Dim fldNomTable As Field
    T = Timer 'juste pour le plaisir de compter le temps que cela prend!
    'Pour ouvrir virtuellement la table utilisée par le formulaire en cours, il faut d'abord définir
la base de donnée que l'on souhaite ouvrir (le 99% du temps il s'agit de la courante)
    Set dbs = CurrentDb' 'Evidemment si ce sont des tables externes liées il faudra mettre le
chemin du fichier *.mdb
    'On peut afficher dans la fenêtre d'exécution chaque champ de la table
    Set tdfNomTable = dbs.TableDefs!NomTable
    For Each fldNomTable In tdfNomTable.Fields
        Debug.Print fldNomTable.Name
    Next fldNomTable
    'Maintenant on souhaite ajouter une valeur dans un des champs
    de la table donc on ouvre virtuellement cette dernière
    Set rstNomLigne = dbs.OpenRecordset("NomTable")
    rstNomLigne.AddNew
    'Attention ici il faut faire attention selon la table à remplir chacun des champs du recordset
    rstNomLigne!NomChamp = NouvelleDonnee
    rstNomLigne.Update
    'Si la table se trouve dans un sous-formulaire il faut alors faire un refresh du formulaire
…'principal en utilisant NomDuFormulaire.Refresh
    msgbox "Routine exécutée en: " & Timer - T & " s"
    'Maintenant on souhaite lire un des champs de la table ligne par ligne
    Set Rs = dbs.OpenRecordset("tblLangues", dbOpenTable)
    Rs.MoveFirst
    Do Until Rs.EOF
        MsgBox Rs![NomDuChamp]
        'ou msgbox Rs.Fields(NomDuChamp)
        Rs.MoveNext
    Loop
End Sub
```

Si vous souhaitez faire des importations dans MS Word ou tout autre programme MS Office, vous pouvez également utiliser les propriétés, objets et méthodes DAO de la même façon que vous utilisez les propriétés, objets et méthodes Microsoft Word et que vous leur faites référence. Une fois que vous avez créé une référence à la bibliothèque d'objets DAO, vous pouvez ouvrir des bases de données, créer et exécuter des requêtes pour extraire un ensemble d'enregistrements et renvoyer les résultats à Word.

Avant de pouvoir utiliser DAO, vous devez créer une référence à la bibliothèque d'objets DAO. Pour ce faire, effectuez la procédure suivante.

Activez Visual Basic Editor dans MS Word par exemple. Ensuite, dans le menu *Outils*, cliquez sur *Références*. Dans la zone Références disponibles, cliquez sur Microsoft DAO 3.6 Object Library.

Si Microsoft DAO 3.6 Object Library n'est pas affiché dans la zone Références disponibles, exécutez le programme d'installation d'Office Professional pour installer Data Access Objects pour Visual Basic. La bibliothèque d'objets Microsoft DAO 3.6 Object Library n'est pas incluse dans la version autonome de Word ou dans l'édition Microsoft Office 2000 Standard.

Cet exemple montre comment ouvrir la base de données depuis MS Word et insérer les éléments de la table *tblArticles* dans le document actif.

```
Sub EcrireAccess()
   'N'oubliez pas d'ajouter la Microsoft D.A.O 3.6 (et non pas la Access Object Library!)
   Set Db = DBEngine.OpenDatabase(Name:="C:\" & "base.mdb")
   Set Rs = Db.OpenRecordset(Name:="tblPays")
   'Attention Microsoft recommande lorsqu'on utilise RecordCount de faire un move last
avant (rs.MoveLast) et de refaire un MoveFirst juste après
   For I = 0 To Rs.RecordCount - 1
      Selection.InsertAfter Text:=Rs.Fields(1).Value
      Rs.MoveNext
      Selection.Collapse Direction:=wdCollapseEnd
      Selection.InsertParagraphAfter
   Next I
   Rs.Close
   Db.Close
   Set Rs. = Nothing
   Set Db = Nothing
End Sub
```

Utilisez la méthode OpenDatabase pour établir une connexion à une base de données et l'ouvrir. Une fois la base de données ouverte, utilisez la méthode OpenRecordset pour accéder à un tableau ou exécuter une requête. Pour vous déplacer dans l'ensemble d'enregistrements, utilisez la méthode Move. Pour rechercher un enregistrement donné, utilisez la méthode Seek (cette méthode ne fonctionne pas sur une table liée). Si vous n'avez besoin que d'un sous-ensemble d'enregistrements et non de la totalité de l'ensemble d'enregistrements, utilisez la méthode CreateQueryDef pour créer une requête personnalisée afin de sélectionner les enregistrements répondant à vos critères. Lorsque vous avez fini d'utiliser une base de données, nous vous conseillons de la fermer en utilisant la méthode Close, afin d'économiser de la mémoire.

Autre exemple qui consiste à transformer en majuscules les caractères d'un champ de donnée nommée "Catégories" d'une table nommée "Employés":

```
Sub Majuscules()
   Dim bd As Database
   Dim rs As DAO.Recordset
   Set bd = CurrentDB 'Evidemment si ce sont des tables externes liées il faudra mettre le
chemin du fichier *.mdb
   Set rs=bd.OpenRecordset("Employés", dbOpenTable)
   Rs.MoveFirst
```

```
    Do Until rs.EOF
        Rs.edit
        Rs![catégorie]=Ucase(rs![catégorie])
        Rs.update
        Rs.MoveNext
    Loop
    Rs.Close
End Sub
```

Autre exemple consistant à mettre à jour certains enregistrements en utilisant une requête pour les sélectionner. Le code est à affecter à un bouton de commande. Il permet d'augmenter le salaire d'une catégorie d'employés d'un certain pourcentage. La catégorie et le pourcentage d'augmentation sont sélectionnés dans deux zones de liste modifiable du formulaire (ce dernier est bien entendu référencé par le mot-clé *Me*). Une requête paramétrée "Employés d'une catégorie" permet de sélectionner les enregistrements à mettre à jour.

```
Sub MiseAJourAvecQuery()
    Dim bd As Database
    Dim rs As Recordset
    Dim Q As QueryDef
    Set bd = CurrentDb
    If IsNull (Me![lstCatégorieAugmentation]) Or IsNull(Me![lstPourcentage]) Then
        MsgBox "Vous devez sélectionner une catégorie et un pourcentage", , "Attention erreur"
    Else
        Set q = bd.QueryDefs("Employés d'une catégorie")
        q.Parameters(0)=Me![lstCatégorieAugmentation]
        Set rs = q.OpenRecordset()
        Rs.MoveFirst
        'La structure For I=0 To… ne fonctionne pour les requêtes à priori donc on passe par un
Do Until pour lire la requête ligne par ligne
        Do Until rs.EOF
            Rs.Edit
            Rs![Salaire] = rs![Salaire].*(1+Me![lstPourcentage])
            Rs.update
            Rs.MoveNext
        Loop
        Rs.close
    End if
End Sub
```

19.30.1.2 Mise à jour de tables DAO

Créez le code V.B.A suivant qui met tous les noms des clients de la table *tblClients* en majuscules:

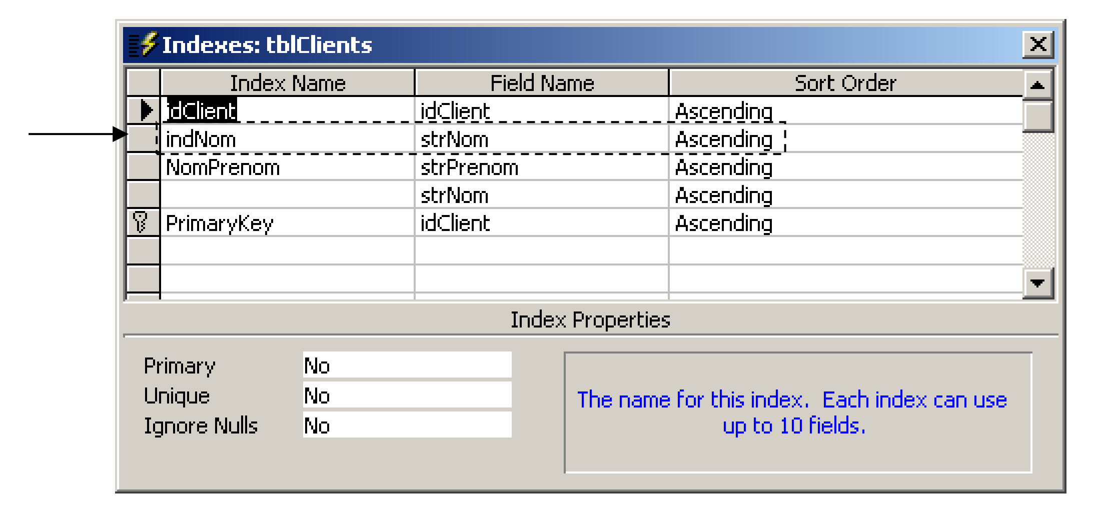
```
Sub TransformMinMaj()
    'Ce programme transforme les noms des clients de la table clients en majuscule
    Dim dbs As Database
    Dim rstCli As DAO.Recordset
    Set dbs = CurrentDb
    Set rstCli = dbs.OpenRecordset("tblClients", dbOpenTable)
```

```
    rstCli.MoveFirst
    With rstCli
        Do Until .EOF
            .Edit
            rstCli("strNom") = UCase(rstCli("strNom"))
            .Update
            .MoveNext
        Loop
    End With
End Sub
```

19.30.1.3 Recherche VBA DAO

Encore plus puissant l'exemple suivant va vous faire comprendre la troisième utilité des Index que nous avions vus dans le cours avancé. Soit la table *tblClients*, nous aimerions créer un moteur de recherche qui retourne le prénom du client (c'est un exemple simple mais extensible à quelque chose de beaucoup plus puissant assez facilement).

D'abord, la fonction .Seek de VBA utilise les index sur les clés de recherche pour fonctionner. Ainsi, si nous souhaitons à partir du Nom d'un client de la table *tblClients* renvoyer le Prénom (*strPrenom*). Ainsi, il nous faut créer un index sur le champ nom que nous nommerons pour l'occasion *indNom*:



Et voici le code:

```
Sub RechercheInfo1()
    'Ce programme recherche une info
    Dim dbs As Database
    Dim rstCli As DAO.Recordset
    Dim strCodeCli As String
    Set dbs = CurrentDb
    Set rstCli = dbs.OpenRecordset("tblClients", dbOpenTable)
    With rstCli
        strNomClient = InputBox("Saisissez le nom du client")
        .Index = "indNom"
        .Seek "=", strNomClient
        If .NoMatch Then
```

```
        MsgBox "Client non trouvé"
    Else
        MsgBox "Prénom: " & rstCli("strPrenom")
    End If
  End With
  rstCli.Close
End Sub
```

Nous pouvons faire également un autre moteur de recherche en utilisant la méthode *Find*:

```
Sub RechercheInfo2()
  Dim dbs As DAO.Database
  Dim rstCli As DAO.Recordset
  Dim strsoc As String
  Dim strCrit As String
  Set dbs = CurrentDb
  Set rstCli = dbs.OpenRecordset("tblClients", dbOpenSnapshot)
  With rstCli
    'saisi du nom client
    strsoc = InputBox("Saisissez les 1ère lettres de du nom du client")
    .MoveLast
    'Recherche
    strCrit = "strNom Like '" & strsoc & "*'"
    .FindFirst strCrit
    'Enregistrement non trouvé
    If .NoMatch Then
      MsgBox "Client non trouvé"
    Else
      'Si trouve: recherche des suivants
      Do While True
        MsgBox rstCli("strNom")
        .FindNext strCrit
        If .NoMatch Then Exit Do
      Loop
    End If
  End With
End Sub
```

19.30.1.4 Exécution requête d'action VBA DAO

Il arrive parfois qu'on doive exécuter une requête d'action de types mise-à-jour, suppression ou création depuis MS Word (typiquement pour le publipostage puisque Word ne peut se lier à une requête) ou MS Excel. Dès lors le code à implément est au minimum le suivant (ici il s'agit de requête d'action de création de table dont il faut supprimer l'ancienne verions au préalable):

```
Sub updatequery()
  Dim db As Database
  Dim query1 As QueryDef
  Set db = OpenDatabase("c:\test.mdb")
  db.Execute "DROP TABLE BackUp"
```

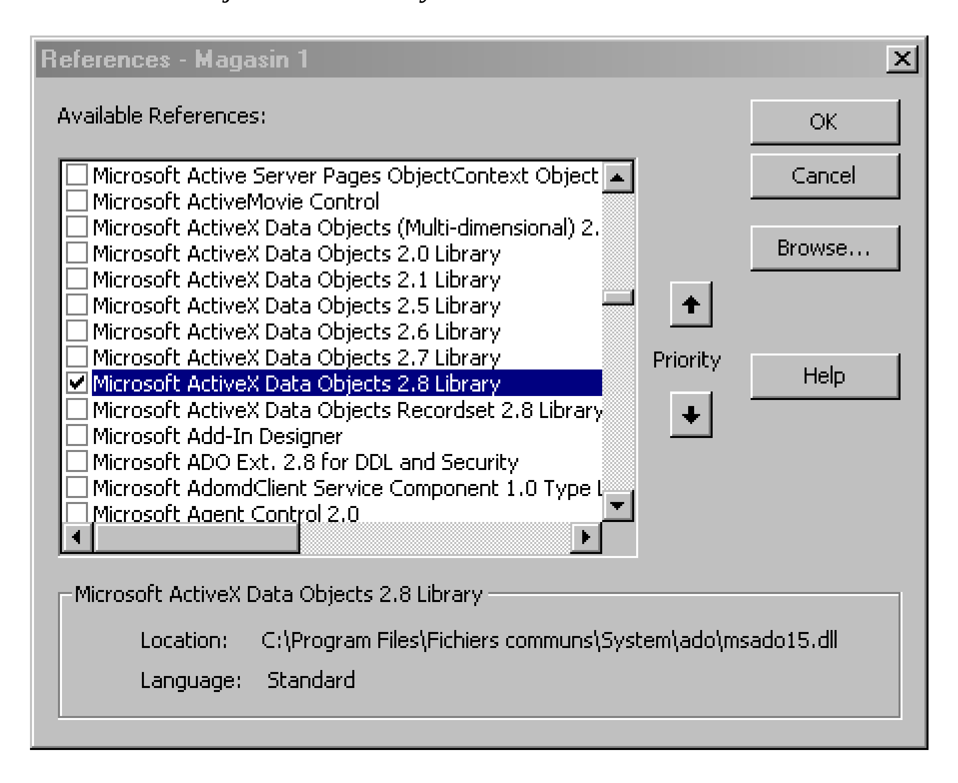
```
    Set query1 = db.QueryDefs("qryCreateBackUp")
      'Sets reference to query name specified
    query1.Execute 'runs query
    Set db = Nothing
    Set query1 = Nothing
End Sub
```

19.30.2 A.D.O

La bibliothèque d'objets A.D.O. vous permet d'écrire une application qui accède à des données situées sur un serveur de bases de donénes et de manipuler celles-ci par les biais d'un fournisseur OLE DB.

ADO a l'avantage d'être facile d'emploi, performant et d'utiliser peu de mémoire et d'espace disque. ADO offre les fonctionnalités de base permettant de créer des applications client/serveur et des applications Web.

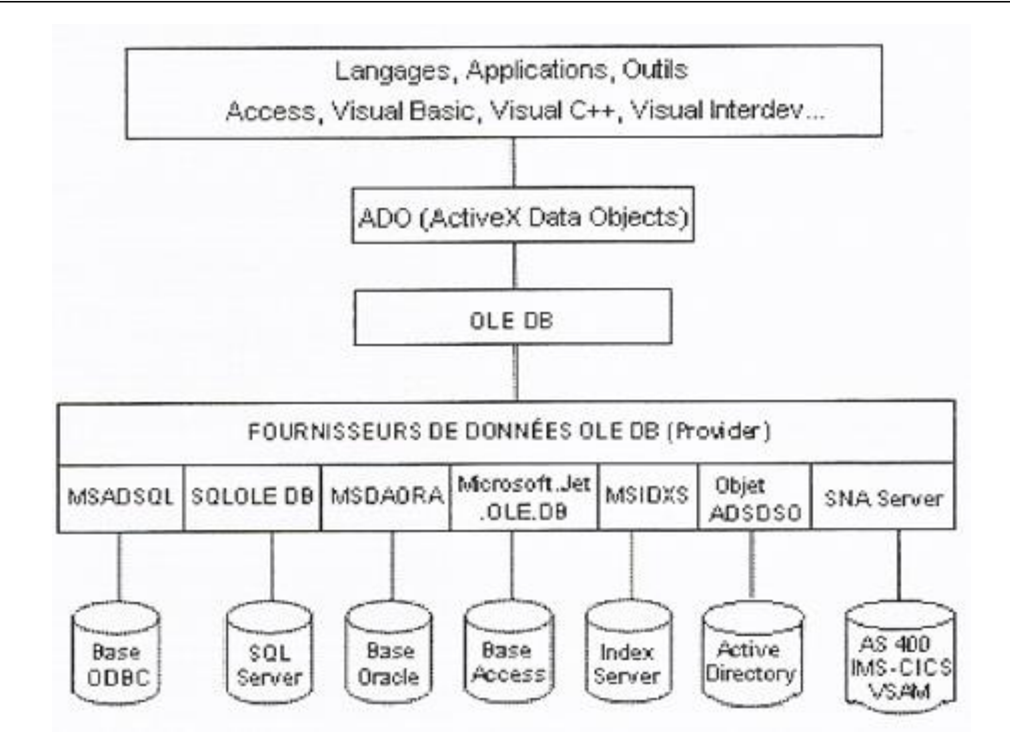
Pour pouvoir utiliser la bibliothèque d'objets ADO, il est nécessaire de cocher la référence *Microsoft ActiveX Data Objects 2.x Library*:



dans la liste des références aux bibliothèques d'objets. Si cette référence n'est pas disponible, vous devez sélectionner le fichier MSA-DO15.dll à parti du bouton de commande *Parcourir*.

OLE DB est une technologie permettant un accès uniforme aux données stockées dans diverses sources d'informations: bases de données relationnelles ou non, messageries, systèmes de fichiers… Quasiment toutes les données de l'entreprise sont accessibles par OLE DB.

ADO est l'interface de référence de programmation utilisé pour accéder aux données de tous les fournisseurs de données OLE DB:



Exemples:

Voici le morceau de code nécessaire à la création d'une référence sur la base de données courante en ADO:

```
Dim cncMagasin as ADODB.Connection
Set cncMagasin = CurrentProject.Connection
```

à une base de données MS Access:

```
Dim cncMagasin as ADODB.Connection
Set cncMagasin = New ADODB.Connection
cncMagasin.Open Provider=Microsoft.Jet.OLEDB.4.0.; Data Source = C:\Magasin.mdb
```

à une base de données SQL Server

```
Dim cnn as ADODB.Connection
cnn.Open Provider=SQLOLEDB.1;Data Source=SRV; Initial Catalog=NomBase;
Integrated Security=SSPI; Persist Security Info=False
```

Voici un exemple plus étendu où nous allons dans notre base de données *Magasin* changer le champ *strPays* de la table *tblClients* pour mettre tous les "France" à la valeur "Suisse":

```
Sub MiseAJour()
   Dim cnc as ADODB.Connection
   Dim strSQL as String
   Set strMessage As String
   Set cnc = CurrentProject.Connection
   StrSQL = "UPDATE tblClients SET strPays = 'Suisse' WHERE strPays='France' "
   cnc.Execute strSQL
   'Demande à l'utilisateur s'il veut confirmer la mise à jour
```

```
  If MsgBox("Confirmez-vous la mise à jour des pays ?", vbYesNo) = vbYes Then
    'Enregistre les modifications
    cnc.CommitTrans
  Else
     'Annule les modification
     cnc.RollbackTrans
  End If
End Sub
```

Le but des exercices qui vont suivre sera de faire communiquer MS Excel et MS Access dans un sens unique et ensuite par complexité croissante, dans les deux sens.

19.30.2.1 MS Excel - lecture ADO

Créez un bouton dans MS Excel exécutant le code suivant (qui se trouvera donc dans un module de MS Excel), qui va chercher les noms et prénoms de la table clients, qui les concatène et les écrits dans la colonne A de la feuille en cours:

```
Sub LireDonnees()
  Dim ADOCnn As ADODB.Connection, ADOTab As ADODB.Recordset
  Dim Index As Long
  Set ADOCnn = New ADODB.Connection
  ADOCnn.Provider = "Microsoft.Jet.OLEDB.4.0"
  'On ouvre la base de données clients
  ADOCnn.Open ("c:/Magasin.mdb")
  Set ADOTab = New ADODB.Recordset
  'On ouvre la base table tblClients de la base de données Magasin.mbd
  'Si on a des besoins basiques, on peut omettre les pointeurs adOpenDynamic et
adLockOptimistic
  ADOTab.Open "tblClients", ADOCnn, adOpenDynamic, adLockOptimistic
  Do While ADOTab.EOF = False
    Index = Index + 1
    Cells(Index, 1) = ADOTab!strPrenom & " " & ADOTab!strNom
    ADOTab.MoveNext
  Loop
  ADOCnn.Close
  Set ADOCnn = Nothing
End Sub
```

19.30.2.2 Curseurs adLockReadOnly, adLockPessimistic, adLockOptimistic, adLockBatchOptimistic

Intéressons nous aux curseurs mentionnés dans le titre que vous avez rencontré dans le code précédent et donnons en une définition.

```
Sub LireDonnees()
  Dim ADOCnn As ADODB.Connection, ADOTab As ADODB.Recordset
  Dim Index As Long
  Set ADOCnn = New ADODB.Connection
  ADOCnn.Provider = "Microsoft.Jet.OLEDB.4.0"
```

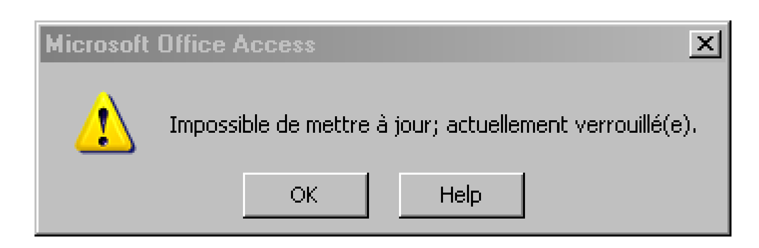
```
    'On ouvre la base de données clients
    ADOCnn.Open ("c:/Magasin.mdb")
    Set ADOTab = New ADODB.Recordset
    'On ouvre la base table tblClients de la base de données Magasin.mbd
    ADOTab.Open "tblClients", ADOCnn, adOpenDynamic, adLockOptimistic
    msgbox
    Do While ADOTab.EOF = False
        Index = Index + 1
        ADOTab!strPrenom = Vincent
        ADOTab.MoveNext
    Loop
    ADOCnn.Close
    Set ADOCnn = Nothing
End Sub
```

Et mettez un point d'arrêt sur la ligne suivante:

```
Do While ADOTab.EOF = False
    Index = Index + 1
   ADOTab!strPrenom = Vincent
    ADOTab.MoveNext
Loop
ADOCnn.Close
Set ADOCnn = Nothing
```

Avec **adLockOptimistic** lorsque le code est arrêté à un certain enregistrement (correspondant à un certaine ligne dans la table *tblClients*) à l'aide du point d'arrêt (simulant un long traitement/calcul), les utilisateur peuvent quand même modifier aussi le champ prénom et valider la ligne. Mais le code plantera car il y aura conflit (il faudra donc faire de la gestion d'erreurs) et ce même si un autre champ le prénom est modifié! Il s'agit également du type de curseur pris par défaut par Access quand aucun n'est spécifié.

Avec **adLockPessimistic** lorsque le code est arrêté à un certain enregistrement (correspondant à un certaine ligne dans la table *tblClients*) à l'aide du point d'arrêt (simulant un long traitement/calcul), les utilisateur peuvent quand même modifier aussi le champ prénom mais pas valider la ligne. Le message suivant leur apparaîtra à l'écran:



Donc l'utilisateur devra annuler ses modifications (avecla touche ESCdu clavier) sur la ligne que le code est entraint de traiter. **adLockPessimistic** est donc un mode exclusif pour le code VBA

Avec **adLockReadOnly** le code VBA ne marchera simplement pas car ce curseur ne permet que la lecture d'informations par le code et non pas la modification. Les utilisateurs peuvent bien évidemment faire n'importe quelle modification et la valider lorsque le code ne fait que de lire des informations.

Avec **adLockBatch** les mises à jour et blockages des lignes ne sont font qu'une fois le code totalement terminé avec une commande spéciale.

19.30.2.3 MS Excel - Recherche ADO

L'objectif de cet exercice est de voir comment chercher des données dans la même base Magasin.mdb. Pour ce faire, nous demandons dans une cellule d'une feuille MS Excel (la cellule C2), le nom que nous recherchons, pour en ressortir le prénom:

```
Sub ChercherDonnees()
   Dim ADOCnn As New ADODB.Connection, ADOTab As New ADODB.Recordset
   ADOCnn.Provider = "Microsoft.Jet.OLEDB.4.0"
   ADOCnn.Open ("c:/Magasin.mdb")
   'Si on a des besoins basiques, on peut omettre les pointeurs adOpenDynamic et
adLockOptimistic
   ADOTab.Open "tblClient", ADOCnn, adOpenDynamic, adLockOptimistic
   With ADOTab
      .Find "Name Like '*" & Range("C2") & "*'"
      If .EOF Then
         MsgBox "Aucune donnée trouvée", vbInformation
      Else
         Range("C2") = !strNom & " " & !strPrenom
      End If
   End With
   ADOCnn.Close
   Set ADOCnn = Nothing
End Sub
```

19.30.2.4 MS Excel - Écriture ADO

Le but ici est similaire (toujours avec la même base, toujours avec la même table) mais d'insérer des données (Nom dans la cellule C6, Prénom dans la cellule C7) dans la table MS Access depuis MS Excel:

```
Sub InsererDonnes()
   Dim ADOCnn As New ADODB.Connection, ADOTab As New ADODB.Recordset
   ADOCnn.Provider = "Microsoft.Jet.OLEDB.4.0"
   ADOCnn.Open ("c:/Magasin.mdb")
   'Si on a des besoins basiques, on peut omettre les pointeurs adOpenDynamic et
adLockOptimistic
   ADOTab.Open "tblClients", ADOCnn, adOpenDynamic, adLockOptimistic
   If (Range("C6") = "" Or Range("C7") = "") Then
      MsgBox "Veuillez saisir un nom ou un prénom S.V.P.", vbInformation
   Else
      ADOTab.AddNew
      ADOTab!strNom = Range("C7")
      ADOTab!strPrenom = Range("C6")
      ADOTab.Update
   End If
   ADOCnn.Close
```

```
    Set ADOCnn = Nothing
End Sub
```

19.30.2.5 MS Word - Recherche ADO

Voici un exemple très important d'utilisation d'ADO dans les entreprises!:

Il est donc possible d'aller chercher des données dans MS Access depuis MS Word et de les insérer à l'emplacement de signets placés dans MS Word. Cette possibilité étant souvent utilisée pour générer des documents complexes et beaucoup plus élaborés que les pauvres rapports de MS Access…

Voici le code permettant de se connecter à notre base *Magasin.mdb* et qui peut aller chercher le nom, prénom et adresse de la table *tblClients* à partir d'une inputbox et les insérer dans trois signets Word nommés respectivement *Adresse*, *Anrede*, *Salutations*:

```
Sub RechercheAdresses()
    Dim ADOCnn As ADODB.Connection, ADOTab As ADODB.Recordset
    Dim Name As String
    Set ADOCnn = New ADODB.Connection
    Name = InputBox("Avec quel nom souhaitez vous remplir le document?", "Recherche
d'Adresse", "Boltzmann")
    If Name > "" Then
        ADOCnn.Provider = "Microsoft.Jet.OLEDB.4.0"
        ADOCnn.Open ("c:\Magasin.mdb")
        Set ADOTab = New ADODB.Recordset
        With ADOTab
    'Si on a des besoins basiques, on peut omettre les pointeurs adOpenDynamic et
adLockOptimistic
            .Open "tblClients", ADOCnn, adOpenDynamic, adLockOptimistic
            .Find "strNom='" & Name & "'"
            If Not .EOF Then
                ActiveDocument.GoTo(What:=wdGoToBookmark, Name:="Adresse").Select
                Selection.InsertAfter .Fields("strNom") & " " & .Fields("strPrenom") & vbCr
                Selection.InsertAfter .Fields("strRue") & vbCrLf
                Selection.InsertAfter .Fields("strPays") & "-" & .Fields("strNPA") & " " &
.Fields("strCanton")
                ActiveDocument.GoTo(What:=wdGoToBookmark, Name:="Titre").Select
                Selection.InsertAfter "Madame, Monsieur" & ","
                ActiveDocument.GoTo(What:=wdGoToBookmark, Name:="Salutations").Select
                Selection.InsertAfter "Madame, Monsieur" & ","
            Else
                MsgBox "Aucune donnée trouvée!", vbCritical
            End If
            .Close
        End With
        ADOCnn.Close
        Set ADOCnn = Nothing
    End If
End Sub
```

Évidemment il est possible de faire beaucoup mieux avec des connaissances de VBA supplémentaires (suivre un cours MS Word ou VBA pour cela).

19.31 DLookup syntaxe

La fonction RechDom (DLookup) est une fonction très fréquemment utilisée en VBA pour rechercher la valeur d'un champ particulier qui fait partie d'un jeu défini d'enregistrements.

Par exemple, supposez que vous disposez d'un formulaire *DétailsCommandes* fondé sur une table *Détails Commandes* et dont la zone de texte *Référence produit* affiche le champ *Réfproduit*. Pour rechercher *Nom du produit* dans une table *Produits* à partir de la valeur de la zone de texte *Réfproduit*, vous pouvez créer une autre zone de texte et affecter l'expression suivante à sa propriété (regardez bien ou sont les guillemets et les apostrophes!):

1er exemple:

```
   Dim Result as variant
   Result = DLookup("NomProduit", "Produits", "Réfproduit ='" &
Forms!DétailsCommandes!Réfproduit & "'")
```

2ème exemple:

```
   Dim Result as variant
   intSearch = 1
   Result = DLookup("[CompanyName]", "Shippers", "[ShipperID] = " & intSearch)
```

3ème exemple:

```
   Dim Result as variant
   Result = DLookup("FieldName","TableName","Criteria=4124")
```

4ème exemple:

```
   Dim Result as variant
   Result = DLookup("FieldName","TableName", "Criteria='salut'")
```

5ème exemple:

```
    Dim Result as variant
   Result = DLookup("FieldName","TableName", "Criteria= #31-12-2001 17:55:32#")
```

6ème exemple:

```
    Dim Result as variant
   Result = DLookup("FieldName","TableName", "Criteria=" &
Forms!FormName!ControlName)
```

7ème exemple:

```
    Dim Result as variant
   Result = DLookup("FieldName","TableName","Criteria= '" &
Forms!FormName!ControlName & "'")
```

8ème exemple:

```
   Dim Result as variant
   Result = DLookup("FieldName","TableName","Criteria= " &
Format(Forms!FormName!ControlName, "\#mm-dd-yyyy hh:mm:ss\#"))
```

9ème exemple:

```
   Dim Result as variant
   Result = DLookup("FieldName","TableName","Criteria1= " &
Forms!FormName!Control1 & " AND Criteria2 = '"& Forms!FormName!Control2 & "'" & "
AND Criteria3 = #" & Forms!FormName!Control3 & "#")
```

Les syntaxes ci-dessus ont été prises sur le site web www.mvps.org (heureusement qu'il y en a qui ont cherché avant nous…)

Créez maintenant dans la base *Magasin.mdb* une table *tblCrediteurs* qui sera une copie de la table clients au niveau de la structure des champs mais videz-y les enregistrements existants.

Créez une fomulaire *frmCrediteurs* permettant de remplir cette table de deux manières:

1. Dans une liste déroulant la possibilité de choisir comme créditeur un individu qui est client du magasin ou de saisir dans cette liste même un individu qu'un n'est pas client

2. Si le créditeur est un client, tous les champs correspondants doivent se remplier respectivement à ce qui est indiqué dans la table *tblClients* sinon quoi l'utilisateur doit pouvoir saisir les informations du créditeur.

19.32 États

Si vous ouvrez la base *Magasin.mdb* il y a un état normalement qui ressemble grosso modo à à la figure représentée ci-dessous:

Articles Microsoft Certified Solution Provider

N° Article	Désignation	N° du fournisseur	Quantité par unité de commande	Prix unitaire
GEN-001	Stylos Bic	2	50	SFr. 0.20
GEN-005	Post-It Notes 657	1	60	SFr. 10.40
GEN-006	Stylos rouges	2	100	SFr. 0.50
GEN-007	Stylos bleu	1	100	SFr. 0.70
GEN-009	Equerre	1	20	SFr. 1.50
INF-001	Tambour	2	5	SFr. 249.00
INF-004	Toner		20	SFr. 85.90
INF-005	Disquette (3.5")	2	1000	SFr. 1.30
INF-007	Plumes	2	5	SFr. 0.12
INF-008	Etiquettes Laser (25 feuilles)	2	10	SFr. 35.90
INF-009	T 60 cm	2	300	SFr. 45.00

mardi, 29. octobre 2002 Page 1 sur 1

Ce qui est regrettable dans cet état c'est que les enregistrements (un par ligne) ne sont pas distinctivement séparés par des couleurs par exemple (c'est le cas le plus fréquent).

Si vous passez en mode création vous avez une ligne de section nommée *Détail*. Double cliquez sur cette barre pour en afficher les propriétés. Dans l'onglet des événements activez *Sur impression* et sélectionnez l'option *Générateur de code*.

Saisissez-y le code suivant (attention il arrive parfois dans Access que si l'on ne passe pas par la méthode indiquée dans le paragraphe suivante le code ne marche pas):

```
Sub Détail_Print(Cancel As Integer, PrintCount As Integer)
   Const conltGray = 13816530
   If Détail.BackColor = conltGray Then
      Détail.BackColor = vbWhite
   Else
      Détail.BackColor = conltGray
   End If
End Sub
```

A ma grande stupéfaction et sans pouvoir donner d'explication, Microsoft à décider de différencier le français et l'anglais lors de la manipulation d'états. Comme vous pouvez le voir "Détail" comporte un accent aigu !!!

19.33 Requêtes

Voyons quelques exemples VBA de ce qu'il est possible de faire avec les requêtes:

19.33.1 Lecture du contenu d'une Query DAO

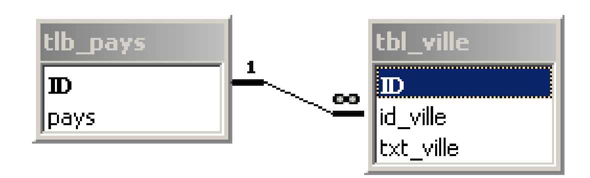
Le code ci-dessous va compter combien d'enregistrements il y a dans une query et lire un des champs de la query un par un:

```
Sub majuscules()
   nb = DCount("*", "qrySommeSorties")
   MsgBox nb
   Dim qdf As DAO.QueryDef
   Dim rcs As DAO.Recordset
 'référence à la requête
   Set qdf = CurrentDb.QueryDefs("qrySommeSorties")
   Set rcs = qdf.OpenRecordset
   rcs.MoveFirst
   Do Until rcs.EOF
      MsgBox rcs![tblArticleNb]
      rcs.MoveNext
   Loop
 'libération de la référence
   Set qdf = Nothing
End Sub
```

19.33.2 ReQuery

Nous allons ici limiter le contenu d'une liste (combo ou zone de liste) à partir du contenu d'une autre liste (utilisation du SQL et du VBA) sans utiliser les macros requery (dont nous avions fait usage lors de notre étude des requêtes).

Soit les tables liées selon le schéma:



où: ID=AutoNum, pays=texte, idVille=nombre, strVille=texte

Considérons maintenant le formulaire suivant (le champ de type "combo" nommé "Pays" est un "LookUp" simple):



L'objectif est d'adapter le contenu du combo Ville en fonction du choix du pays.

Pour ce faire, il suffit de créer un liste combo simple depuis la barre d'outils contrôle (relié à aucune table, à aucune requête ou autre).

Supposons que:

1. le champ de liste combo "Pays" se nomme "strPays"

2. le champ de liste combo "Ville" se nomme "strVille"

Dans le procédure événementielle "Sortie" du champ de liste combo "Pays", insérez le code suivant (pour la procédure SQL on peut s'aider d'une requête qui affiche les villes en fonction du paramètre "id_ville" puisque c'est ce que retourne le champ de liste modifiable pays):

```
Sub strPays_Exit(Cancel As Integer)
   Dim strSQL As String
   strSQL = "SELECT tbl_ville.strVille FROM tblPays INNER JOIN tblVille ON tblPays.ID
= tblVille.idVille WHERE (((tblVille.idVille)=" & Me!strPays
   strSQL = strSQL & "))"
   Me!strVille.RowSourceType = "Table/Query"
   Me!strVille.RowSource = strSQL
   strSQL = "SELECT strPays FROM tblPays"
   Me!strPays.RowSourceType = "Table/Query"
   Me!strPays.RowSource = strSQL
   MsgBox "L'ajout a bien été effectué. Veuillez maintenant le saisir dans la liste"
End Sub
```

Ce système est généralisable à n'importe quoi mais ne fonctionne pas avec les champs de type texte !!! Pour sortir une information unique dans un champ de formulaire à partir du script précédent, il faut utiliser une zone de liste et utiliser la procédure événementielle "Sur changement".

Maintenant que nous avons une petite idée de comment faire du VBA et du SQL poussons un petit peu plus loin le sujet pour créer des formulaires de recherche multi-critères (concept très important dans Access et non disponible par assistant à ce jour).

19.34 Choix multiple

Le but de l'exercice est le suivant: selon l'exemple générique ci-dessous, créez un rapport qui affiche la liste des sorties dans un état simple mais en rajoutant le fait que dans un formulaire, l'utilisateur doit pouvoir dans une liste à choix multiple sélectionner les articles qu'il désire visualiser dans l'état (durée: 1 heure).

```
….
   Dim frm as Form, ctl As Control
   Dim varItem as Variant
   Dim strSQL as string
   Set frm = Form!frmMyForm
   Set ctl= frm!lbMultiSelectListbox
   strSQL="Select * from Employees where [EmpID]="
   'On suppose que la valeur de type long, [EmpID], est le champ contre lequel on se
   'compare aux valeurs choisies dans la liste à choix multiple
   For Each varItem In ctl.ItemsSelected
      strsSQL=srtSQL & ctl.ItemData(varItem) & " OR [EmpID]="
   Next varItem
```

```
   'on enlève le dernier segment
   strSQL=left$(strSQL,len(strSQL)-12))
….
```

19.35 Calendrier

MS Access ne contient toujours pas d'état ou de formulaire adapté à générer un calendrier quel qu'ilsoit. Voici le code pour en générer un à la volée dans votre base:

```
Public Function Getdate() As Variant
   ' edit to get what ever return you want
   Dim stFRmName As String
  stFRmName = CreateCalForm
  Do Until IsDate(Forms(stFRmName).Tag) ' check to see if date selected
    DoEvents
  Loop
  Getdate = Forms(stFRmName).Tag
  DoCmd.Close acForm, stFRmName, acSaveNo
End Function

Public Function CreateCalForm() As String
  Dim frm As Form
  'creates form
  Set frm = CreateForm
  With frm 'set any additional form properties you need here
    .Caption = "POP UP Calender"
    .RecordSelectors = False
    .NavigationButtons = False
    .Width = 2.5
    .Section(acDetail).Height = 3
    CreateCalForm = .Name
  End With
  Call LoadControls(frm)
End Function

Public Function LoadControls(ByVal frm As Form)
  Dim txt(42) As Control, ctlMonth As Control
  Dim ctlYear As Control
  Dim txtLeft As Integer, TxtTop As Integer
  Dim xPos As Integer, yPos As Integer, X As Integer
  Dim TxtHeight As Integer, TxtWidth As Integer
  Dim stFRmNam As String
  stFRmNam = frm.Name
  'Adjust to size boxes. Could base on size of form
  TxtHeight = 1000
  TxtWidth = 1000
  TxtTop = 1000
  txtLeft = 5
  'creates 6x7 grid of text boxes
  For yPos = 1 To 6
    For xPos = 1 To 7
```

```
      Set txt(X) = CreateControl(stFRmNam, acTextBox, , "", "", txtLeft, TxtTop, TxtWidth,
      TxtHeight)
      txtLeft = txtLeft + TxtWidth 'sets with of text boxes
      'set any additional properties or events here
      txt(X).OnClick = "=Clicked()" 'add on click event to each box
      txt(X).Name = "d" & X + 1 'name boxes like an array
      X = X + 1
    Next xPos 'next box
    txtLeft = 5 'go back to left start position
    TxtTop = TxtTop + TxtHeight 'drop down height
  Next yPos  'next row
  'create additional txtboxes to hold Month & year
  Set ctlMonth = CreateControl(stFRmNam, acTextBox, , "", "", 100, 10, 1000, 300)
  ctlMonth.Name = "txMonth"
  Set ctlYear = CreateControl(stFRmNam, acTextBox, , "", "", 1100, 10, 1000, 300)
  ctlYear.Name = "txYear"
  'load module to load dates and open form
  Call LoadModule(frm)
  DoCmd.OpenForm stFRmNam
End Function

Public Function LoadModule(ByVal frm As Form)
  Dim Mdl As Module
  Dim stBld As String
  'creates module
  Set Mdl = frm.Module
  'creates eventprocedures This is ugly to save space. You can see it better
  'Once form is created click on design view
  stBld = stBld & "Private Function LoadCal(dtMonth as integer, dtyear as integer)" &
  vbCrLf
  stBld = stBld & "Dim Curday as Variant, dtFirst as variant" & vbCrLf
  stBld = stBld & "curday = DateSerial(dtyear, dtmonth, 1)" & vbCrLf
  stBld = stBld & "dtFirst = curday" & vbCrLf
  stBld = stBld & "me![txMonth] = Format(dtFirst, ""m"")" & vbCrLf
  stBld = stBld & "me![txYear] = Format(dtFirst, ""YYYY"")" & vbCrLf
  stBld = stBld & "Do Until curday = DateSerial(dtyear, dtmonth + 1, 1)" & vbCrLf
  stBld = stBld & "'Paramétrer ici -1 pour système américain ou -2 pour européen" & vbCrLf
  stBld = stBld & "me(""D"" & Day(curday) + WeekDay(dtFirst) - 2) = Day(curday) & "" ""
  & Left(Format(Curday, ""DDDD"", vbMonday), 3)" & vbCrLf
  stBld = stBld & "If Weekday(Curday, vbMonday) = 6 Or Weekday(Curday, vbMonday) = 7
  Then" & vbCrLf & "me(""D"" & Day(Curday) + Weekday(dtFirst) - 2).BackColor =
  RGB(255, 255, 0)" & vbCrLf & "Else" & vbCrLf & "Me(""D"" & Day(Curday) +
  Weekday(dtFirst) - 2).BackColor = RGB(255, 255, 255)" & vbCrLf & "End If" & vbCrLf
  stBld = stBld & vbCrLf & "curday = DateAdd(""d"", 1, curday)" & vbCrLf & "Loop" &
  vbCrLf
  stBld = stBld & "End Function" & vbCrLf
  stBld = stBld & "Private Sub Form_Load()" & vbCrLf
  stBld = stBld & "DoCmd.RunCommand (acCmdSizeToFitForm)" & vbCrLf
  stBld = stBld & "Call LoadCal(Month(date),Year(date))" & vbCrLf & " End Sub" &
  vbCrLf
```

```
  stBld = stBld & "Public Function Clicked()" & vbCrLf
  stBld = stBld & "me.tag = txmonth & ""/"" & me.activecontrol.value & ""/"" & txYear" &
   vbCrLf & "End Function"
  stBld = stBld & vbCrLf & "Private Sub txMonth_AfterUpdate()" & vbCrLf
  stBld = stBld & "call ClearCal()" & vbCrLf & "call LoadCal(me!txmonth, me!txYear)" &
  vbCrLf
  stBld = stBld & "End Sub" & vbCrLf
  stBld = stBld & "Private Sub txYear_AfterUpdate()" & vbCrLf
  stBld = stBld & "call ClearCal()" & vbCrLf & "call LoadCal(me!txmonth, me!txYear)" &
  vbCrLf
  stBld = stBld & "End Sub" & vbCrLf
  stBld = stBld & "Private Sub ClearCal()" & vbCrLf
  stBld = stBld & "Dim X as integer" & vbCrLf & "for x = 1 to 42" & vbCrLf
  stBld = stBld & "me(""D"" & x) = """ & vbCrLf
  stBld = stBld & "next x" & vbCrLf & "End Sub"
  'can also use ( insertlines addfromfile addfromstring) in place of inserttext
  Mdl.InsertText stBld
End Function
```

19.36 Connexion

Nous avons vu page 350 comment fractionner une base entre vue logique et utilisateur (tables et interface). Cependant lorsqu'un fichier MDE est envoyé à un client et que l'ensemble des menus et barre d'outils sont bloqués il faut pouvoir par le VBA redéfinir le chemin de connexion des tables à chaque fois au démarrage de la base en fonction du choix du client en ce qui concerne la localisation du fichier contenant les tables.

Si nous supposons que vous ne pouvez intervenir physiquement chez le client ou à distance par prise de contrôle du serveur et que vous ne pouvez laisser faire celui-ci le travail à votre place, la seule manière d'y remédier en connaissant à l'avance le chemin du disque réseau vers le fichier des tables est de lancer le code suivant au démarrage de la base:

```
Sub Attache_Tables()
   Dim db As Database
   Dim TD As TableDef
   Dim i As Integer, Réponse As Variant
   Set db = DBEngine.Workspaces(0).Databases(0)

   On Error GoTo ErrAT

   Reponse = SysCmd(acSysCmdInitMeter, "Attache les tables", db.TableDefs.Count - 1)

   'boucle sur toutes les tables, réattachant celles dont la chaîne de connection est non vide, les
autres étant des tables locales

   For i = 0 To db.TableDefs.Count - 1
      Set TD = db.TableDefs(i)

      ' si c'est une table attachée, on essaie de la réattacher

      If TD.Connect <> "" Then
```

```
        TD.Connect = ";DATABASE=" & ("k:\tables.mdb")
        TD.RefreshLink
        If Err <> 0 Then
          MsgBox TD.Name
        End If
      End If
      Reponse = SysCmd(acSysCmdUpdateMeter, i + 1)
   Next i

   DoCmd.Hourglass False
   Reponse = SysCmd(acSysCmdClearStatus)

   Exit Sub
ErrAT:
   MsgBox "L'application n'a pas pu trouver le fichier tables.mdb sur le serveur K:\." &
Chr(13) & "Veuillez contacter le responsable de la base.", vbCritical + vbOKOnly
End Sub
```

19.37 Nouveautés VBA Access 2007-2010

Nous souhaitons ici donner des indications relativement à des changements de comportement de MS Access 2007 et 2010.

D'abord en ce qui concerne ADODB on ne peut plus coder une connexion comme nous le voulons. Maintenant il est obligé de le faire proprement et d'utiliser le nouveau pilote ACE.OLEDB (qui est le même pour Access 2007 et 2010).

Par exemple:

19.37.1 Connecteur ADODB

```
Sub ConnectionBD()
      Set ADOCnn = CurrentProject.Connection
      Dim ADOTab As ADODB.Recordset
      Set ADOTab = New ADODB.Recordset
     'Si on a des besoins basiques, on peut omettre les pointeurs adOpenDynamic et
adLockOptimistic
      ADOTab.Open "tblLastUpdate", ADOCnn, adOpenDynamic, adLockOptimistic
      Do While ADOTab.EOF = False
         ADOTab.Fields("datLastUpdate").Value = Now
         ADOTab.MoveNext
      Loop
      ADOCnn.Close
      Set ADOCnn = Nothing
      'Nous mettons ensuite la meme information dans le lable apres avoir clique sur le bouton
End Sub
```

19.37.2 Export flat-file (schema.ini)

Avant Access 2007/2010, le logiciel se référait aux paramètres de MS Windows pour les paramètres d'export des fichiers flats files du type *.txt ou *.csv. Ceci n'est donc plus le cas avec les nouvelles versions.

Le développeur doit maintenant obligatoirement (avant c'était conseillé afin d'éviter de mauvais suprises mais pas obligatoire…) créer un fichier *schema.ini* dans le même dossier que là où s'effectuera l'export et contenant les spécifications d'export (il en va de même pour l'import)!

Voici un exemple:

```
Sub SchemaIniExport()
   filSchemaIni = FreeFile
   Open CurrentProject.Path & "\schema.ini" For Output As filSchemaIni
   Print #filSchemaIni, "[" & "MonFichier" & ".csv]"
   Print #filSchemaIni, "ColNameHeader = False"
   Print #filSchemaIni, "CharacterSet = 1252"
   Print #filSchemaIni, "Format=Delimited(;)"
   Close filSchemaIni
   DoCmd.TransferText acExportDelim, "", "NomTableouRequeteAExporter",
CurrentProject.Path & "\MonFichier.csv"
End Sub
```

19.37.3 Mode Hors-Ligne/En-Ligne SharePoint

Une nouveauté majeure depuis Access 2007 est de pouvoir prendre le contenu d'une base de données entièrement basée sur du SharePoint hors-ligne. Seul petit hic, le bouton n'est pas facilement accessible pour un utilisateur non averti. Le mieux est donc de créer un bouton sur le formulaire d'accueil de votre base de données. Ceci est alors facilement faisable en utilisant l'uniquement commande VBA suivante:

```
Access.RunCommand (Access.AcCommand.acCmdToggleOffline)
```

20 Confusions courantes

Dans MS Access suivant le niveau d'utilisation que nous faisons du logiciel, certaines fonctionnalités peuvent paraître faire doublon avec d'autres.

Une analogie de se problème peut être faire avec le tableur MS Excel où finalement la fonctionne MOYENNE ne sert objectivement pas à grand chose puisqu'il s'agit d'une somme divisée par un comptage. Mais dans certaines situations particulières… cela ne convient plus et alors il faut opter pour d'autres solutions. Il en va de même pour MS Access.

J'ai souhaité dans le présent chapitre faire un petit listing avec des captures d'écran des outils que les utilisateurs trouvent très souvent comme faisant doublons avec d'autres existants tout en expliquant le pourquoi du comment.

20.1 Masques VS Valide Si, VBA & Null interdit

Souvent les créateurs de base de données débutants considèrent que dans les tables ou formulaires, les masques des saisies, les contrôles de validation, le null interdit, la chaîne vide autorisée ou le VBA font un peu doublon. Ils ont raison dans certaines situations!

Si nous prenons la situation suivante:

Général | Liste de choix

Taille du champ	7
Format	
Masque de saisie	>LLL\-000
Légende	Article
Valeur par défaut	
Valide si	NbCar([strNbArticle])=7
Message si erreur	
Null interdit	Oui
Chaîne vide autorisée	Non

Donc par exemple notre masque de saisie des codes articles est le suivant:

>LLL-000;0;

Il est clair que jamais un utilisateur ne pourra créer d'articles sans mettre au moins 3 lettres et 3 chiffres. Dès lors vaut-il la peine de mettre un *Valide Si* du type suivant dans le champ de la table:

NBCAR([strNbArticle])=7

La réponse est: **Non**. Quelque soit la situation (macros, VBA, SQL) les deux feront doublon.

Evidemment, les utilisateurs experts en VBA savent que ce langage permet de faire des masques de saisie dans les formulaires beaucoup plus complexes et pertinents que le masque de saisie par défaut intégré dans les tables (ou celui intégré par défaut dans les formulaires). Pour ces utilisateurs, l'utilisation du masque de saisie standard de MS Access est donc un doublon et de plus un outil inutile étant donné le peu de flexibilité dont il dispose.

Enfin, si nous avons le masque de saisie standard ci-dessus ou géré avec du VBA est-il utile de mettre l'option *Null Interdit* dans le champ concerné de la table?

La réponse est: **Cela dépend**.

Effectivement, si vous savez que votre base de données ne sera jamais migrée dans une autre technologie, c'est inutile. Par contre si vous migrez votre base de données MS Access dans Oracle ou SQL Server, tout ce qui est relatif aux masques de saisies ou critières de validations sera supprimé. A l'opposé, les propriétés *Null Interdit* et *Chaîne vide autorisée* sont normalement un standard international dans le monde des bases de données relationnelles.

Enfin, étant donné le masque, il paraît clair que mettre la propriété *Chaîne vide autorisée* à *Non* fait doublon. Qu'en-est-il?

La réponse est: **Cela dépend** et ce pour exactement les mêmes raisons que la propriété *Null Interdit.*

20.2 Valide Si VS VBA & Null interdit

Prenons un exemple classique de validation *Valide Si* qui ne peut pas cette fois faire doublon avec un masque de saisie standard (non VBA):

>=SérieDate(Année(Maintenant());Mois(Maintenant())-2;Jour(Maintenant()))

comme nous pouvons le voir sur la capture d'écran ci-dessous:

Général | Liste de choix

Format	Date, abrégé
Masque de saisie	00.00.0000;0;_
Légende	Date de sortie
Valeur par défaut	
Valide si	>=SérieDate(Année(Maintenant());Mois(Maintenant())-2;Jour(Maintenant()))
Message si erreur	
Null interdit	Non
Indexé	Non
Mode IME	Aucun contrôle
Mode de formulation IM	Aucun
Balises actives	
Aligner le texte	Général
Afficher le sélecteur de c	À certaines dates

La question étant de savoir si cela ferait doublon avec un code VBA qui ferait la même type de contrôle mais dans un formulaire.

La réponse est: **Oui**. Mais le problème des *Valide Si* c'est qu'on ne peut pas le paramètrer dynamiquement à partir de variables qui seraient contenues dans des tables différentes (par exemple ci-dessus le délai de 2 mois devrait pas être écrit en dur!) et que les messages d'erreurs sont catastrophiques en termes de plus value. Donc le VBA reste toutefois indispensable et beaucoup plus flexible et puissant.

Maintenant une autre question se pose. Cette condition de validation semble imposer qu'une date existe. Au fait, il n'en est rien. Donc à la question de savoir si cette validation ferait doublon avec un *Null Interdit* à *Oui* la réponse est **Non**.

Par contre, ceux qui choisiraient de mettre un masque de saisie standard (non VBA) obligatoire pour la date du type:

00.00.0000;0;

et qui auraient activé le *Null Interdit* à *Oui* se trouveraient alors dans une situation de doublon puisque le masque de saisie ne laisserait pas l'utilisateur valider l'enregistrement sans aucune date. Ceci dit, il est quand même nécessaire de mettre le *Null Interdit* à *Oui* si nous savons qu'un jour la base de données MS Access sera migrée sur une autre technologie car les masques n'existent pas dans la majorité des progiciels de type SGBDR.

20.3 Null interdit VS VBA

La question ici est donc de savoir s'il vaut mieux mettre un champ en *Null Interdit* dans une table ou de contrôler avec du VBA qu'un champ ne soit jamais laissé vide dans un formulaire ou… de faire les deux.

La réponse est: **cela dépend**!

Effectivement dans certaines situations où des formulaires utilisent des macros de type *Actualiser* sur des listes déroulantes, l'option *Null Interdit* peut poser parfois des problèmes énormes qu'il faut pouvoir contourner de manière très astucieuse avec du VBA.

Dans des cas simples, le VBA sera préféré car le message d'erreur du *Null Interdit* est une catastrophe et n'apporte aucune information compréhensible à un utilisateur de base.

Le mieux est donc d'y associer les deux et d'empêcher un utilisateur en faisant usage du VBA de sortir d'un champ de saisie tant que ce dernier n'est pas informé et avant que le message d'erreur du *Null Interdit* s'active (ou toute autre technique similaire).

20.4 Intégrité VS Limité à la liste & VBA

Comme nous l'avons vu, l'intégrité référentielle permet de s'assurer de la cohérence des données entre deux tables liées (fonction bijective). Cependant de nombreux participants trouvent que l'activation de l'intégrité référentielle fait doublon avec l'option *Limiter à liste* disponble lorsque la relation est basée sur une liste à choix. D'autres encore trouvent l'un et l'autre inutile puisque l'on peut mettre en place un contrôle de l'intégrité avec le VBA.

La question est de savoir s'il vaut mieux mettre l'option *Limiter à la liste* ou seulement l'intégrité référentielle, ou les deux en même temps, ou s'il vaut mieux tout contrôler en VBA.

La réponse est: **il vaut mieux avoir trop de contrôle que pas assez**

Ainsi, nous conseillons vivement d'activer l'option *Limiter à la liste* et l'option *Intégrité référentielle* afin d'éviter des surprises dans l'évolution future de la base de données au cas où un utilisateur non confirmé reprendrait le flambeau de son développement (il arrive fréquemment qu'ils suppriment les propriétés des listes déroulantes mais très rarement qu'ils désactivent l'option d'intégrité référentielle car ils n'en comprennent pas le sens).

En ce qui concerne le VBA il faut avouer que ce n'est que dans des cas particuliers et complexes qu'on va traiter l'intégrité référentielle avec. Cependant le contrôle d'intégrité sera souvent beaucoup plus lent qu'avec le moteur de base de données d'Access.

20.5 Clés primaires combinées VS Macro Requery & VBA

Nous avons vu plus haut dans le présent ouvrage le concept de clés primaires combinées (ou index combinés) qui permet de s'assurerde l'unicité d'un couple de champ dans une table seule ou dans une table de transition.

D'abord, il est évident encore une fois que les clés primaires combinées (ou index combinées) peuvent être substitués par du code VBA dans le cadre des contrôles de saisie via les formulaires et réduit dès lors la taille de l'index de la base de données. Mais le problème est la compléxité de mise en œuvre du VBA qui nécessite beaucoup plus de temps et de connaissance et aussi le fait que cette approche par le VBA ne fonctionnerait que pour des saisies faites via des formulaires (alors que beacoup d'entreprises n'utilisent pas du tout les formulaires).

Ensuite, en ce qui concerne l'usage d'une macro requery dans le cas des tables de transition en lieu et place des clés primaires combinées (ou index combinées) et du VBA le problème est au fait exactement le même outre la complexité de mise en œuvre.

Le mieux est au fait d'utiliser les trois comme cela se fait dans les bases de données MS Access créées par des professionnels: les clés primaires et index combinées sont dans le modèle physique de la base de données comme première couche de sécurité, ensuite dans les formulaires il y a des macros requery pour les tables de transition à clés primaires ou index multiples et il y a aussi du VBA pour améliorer l'aide et le contrôle et la gestion de l'erreur lors de la saisie.

20.6 État VS Formulaire A4

Dans certaines situations il est possible d'utiliser les formulaires Access en lieu et place des états puisque les formualaires peuvent aussi s'imprimer.

L'avantage d'utiliser les formulaires étant que l'on peut y mettre des boutons, des listes déroulantes et autres éléments interactifs ce qui n'est pas possible sur un état (du moins à ma connaissance).

Un exemple typique est le formulaire de notes de facture. Quand une personne créée une nouvelle facture via le formulaire, elle pourrait aussi l'imprimer en même temps qu'elle fait la saisie (ou plus tard).

Les états (rapports) seront eux plutôt utilisées pour agréger massivement des données sous forme d'un rapport structuré avec des regroupements, une page de garde, des numérotations de pages, etc.

20.7 Sécurité MDW VS Sécurité VBA

La sécurité MDW ayant disparu dans le nouveau format de fichier MS Access 2007 et ultérieur la questio ne se pose plus vraiment. D'autant plus que ceux qui souhaitent rester en *.mdb ou *.mde ne peuvent pas bénéficier de certaines nouveautés très importantes et utiles de MS Access (tout ce qui est relatif à MS SharePoint particulièrement!).

Malheureusement le fait de devoir coder en VBA la sécurité avec des tables multiples contenant les noms des utilisateurs, les groupes de sécurité, les types de sécurité, les objets et

sous-objets est un travail très très très laborieux qui nécessite même sur une petite base de données un petite centaine d'heures de travail.

Ceci dit, pour ceux qui travaillent avec des fichiers *.mdb/*.mde associée à des espaces de travail de sécurité *.mdw, le VBA reste quand même un complément souvent indispensable.

Donc les deux ne font jamais effet de doublon et ce même dans les anciens fichiers.

Enfin pour clore cette partie, il faut savoir qu'avant la 2007 il y avait une tendance à privilégier la sécurité intégrée - quand cela était possible - car elle permettait d'économiser beaucoup de temps pour les développeurs et donc beaucoup d'argent aussi même si elle était moins flexible que le VBA.

21 VBScript

Code pratique en VBScript (cours d'initiation de 2 jours) pour au démarrage de l'ordinateur (au fait l'idée est plutôt d'utiliser les tâches automatiques de Windows pour lancer le script à une heure donnée sinon il suffit de mettre le fichier dans le dossier *Démarrage*...), ouvrir un fichier MS Excel et exécuter une macro spécifique à l'intérieur de celui-ci:

```
Set accDB = CreateObject("Access.Application")
accDB.Application.Visible = True
accDB.OpenCurrentDatabase("c:\db1.mdb")
accDB.DoCmd.RunMacro "Macro1" 'pour lancer une macro
accDB.DoCmd.OpenModule "Module1","test" 'pour lancer une routine VBA
accDB.usercontrol = True 'si on enlève cette ligne Access se ferme
```

Si on veut faire un script qui se lance toutes les minutes:

```
Set accDB = CreateObject("Access.Application")

Do
  WScript.Sleep (60*1000)
  accDB.Application.Visible = False
  accDB.OpenCurrentDatabase("c:\db1.mdb")
  accDB.DoCmd.RunMacro "Macro1" 'pour lancer une macro
  accDB.DoCmd.OpenModule "Module1","test" 'pour lancer une routine VBA
loop
```

Pour arrêter le script dans le gestionnaire des tâches il suffit d'arrêter le process *wscript*

22 XML – XSL

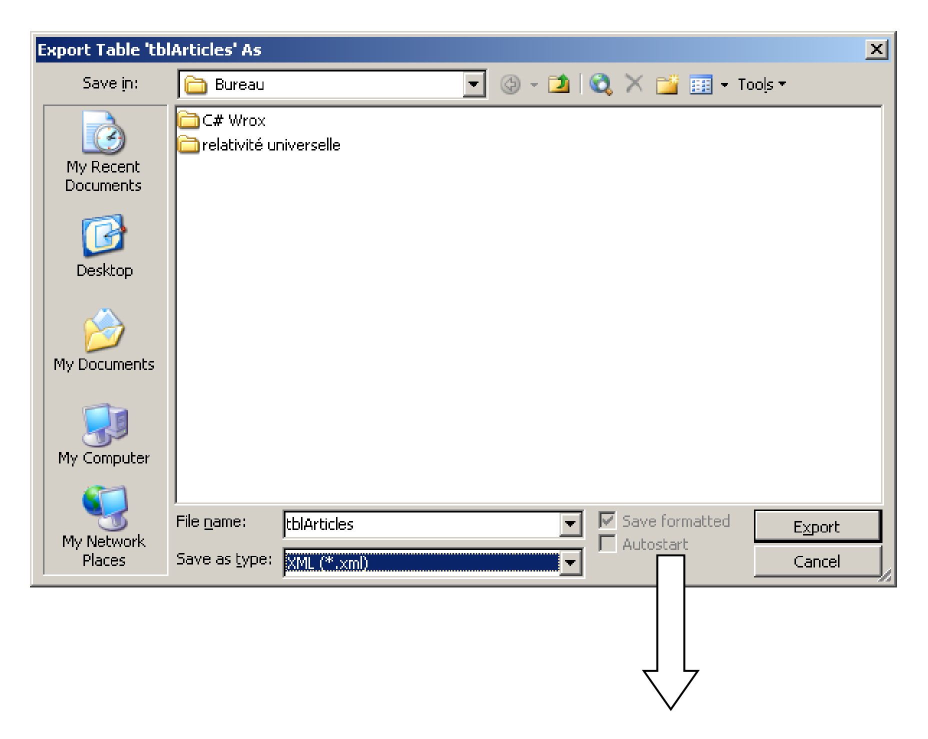
Le XML fait l'objet d'un cours à lui seul (2 jours) où l'on apprend à traiter celui-ci dans les logiciels MS Word 2003 (pour la création de documents), dans MS Excel 2003 (pour la gestion des données) et dans MS InfoPath 2003 (pour la création de formulaires). Cependant, dans le cadre d'un cours MS Access pour développeurs, nous prendrons pour référence le support de cours de "Sup. Info Paris" ou celui de Luc Van Lancker complété par les exercices et la présentation de Vincent Isoz.

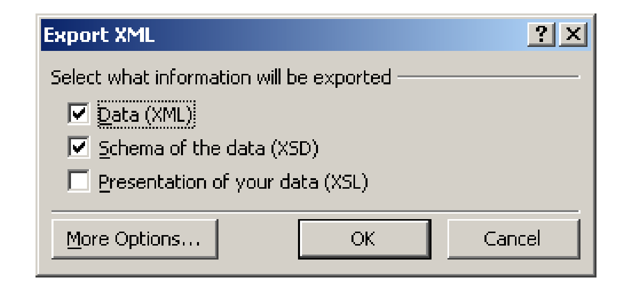
Le méta-langage XML est absolument incontournable aujourd'hui dans les nouvelles technologies s'impose comme standard international dans l'échange des données. Ainsi, MS Access est-il directement concerné. Cette voie vers le XML que prennent tous les logiciels dans le domaines de l'informatique se fait particulièrement ressentir pour les utilisateurs bureautiques avec la version MS Office 2003 où il est omniprésent même dans MS Word et MS Excel. Il convient dès lors d'acquérir des nouvelles méthodes de travail. Malheureuses, l'inertie d'adaptation des utilisateurs moyens fait que probablement cette technologie sera omniprésente chez le grand public vers 2010….

Démonstration de l'utilisation simple XML:

1. Ouvrez une table quelconque dans un base Access (pour l'exemple nous nous sommes servis d'une table nommée *tblArticles*)

2. Allez dans le menu *Fichier/Exporter* et choisissez l'option *XML* (si MS Access vous demande de générer un fichier XSL et XSD… laissez tomber !!! Le code y est catastrophique… et c'est inutilisable):





3. Ouvrez le fichier XML résultant dans Internet Explorer

Voici à quoi ressemblera le résultat:


```
<?xml version="1.0" encoding="UTF-8" ?>
- <dataroot xmlns:od="urn:schemas-microsoft-com:officedata" generated="2004-04-07T07:05:45">
+ <tbl_articles>
+ <tbl_articles>
+ <tbl_articles>
+ <tbl_articles>
+ <tbl_articles>
+ <tbl_articles>
- <tbl_articles>
    <ID_articles>7</ID_articles>
    <txt_nbarticle>INF-002</txt_nbarticle>
    <mem_designation>Disquette (3.5")</mem_designation>
    <int_nbfournisseur>3</int_nbfournisseur>
    <int_moq>1000</int_moq>
    <cur_unitprice>1.3</cur_unitprice>
  </tbl_articles>
- <tbl_articles>
    <ID_articles>8</ID_articles>
    <txt_nbarticle>INF-003</txt_nbarticle>
    <mem_designation>Etiquettes LASER (25 feuilles)</mem_designation>
    <int_nbfournisseur>3</int_nbfournisseur>
    <int_moq>10</int_moq>
    <cur_unitprice>35.9</cur_unitprice>
  </tbl_articles>
- <tbl_articles>
    <ID_articles>10</ID_articles>
    <txt_nbarticle>INF-004</txt_nbarticle>
    <mem_designation>Toner</mem_designation>
    <int_nbfournisseur>3</int_nbfournisseur>
    <int_moq>20</int_moq>
    <cur_unitprice>85.9</cur_unitprice>
  </tbl_articles>
</dataroot>
```


Nous voyons que le code est beaucoup plus propre que celui généré automatique par les autres logiciels de la suite MS Office (ce qui est logique par ailleurs).

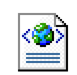
Voici une première méthode pour exporter en XML:

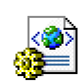
```
Sub ExportTable()

'Ce code va créer un fichier XML, XSL et HTML
Application.ExportXML ObjectType:=acExportTable, DataSource:="tblArticles",
DataTarget:="C:\tblArticles.xml", PresentationTarget:="C:\tblArticles.xsl",
Encoding:=acUTF8

End Sub
```







Pour pouvoir affichier correctement la liste des clients, vous devez soit utiliser le fichier tblArticles.htm qui fait le lien entre les fichiers xml et xsl, soit modifier le fichier tblClients.xml afin de lui associer le fichier tblClietns.xsl.

Une autre manière de générer un fichier XML de façon très modulable à partir de VBA est la suivante (exemple générique pris sur Internet et facilement généralisable):

```
Option Compare Database

Option Explicit

Sub MainXML()
    Dim kvl_tabledef As New TableDef
    Dim kvl_xml As Integer
    Dim kvl_recordset As New ADODB.Recordset
    Dim kvl_sql As String
    Set kvl_tabledef = CurrentDb.CreateTableDef("stefan_ram_table")
    kvl_tabledef.Fields.Append kvl_tabledef.CreateField("name", dbText)
    kvl_tabledef.Fields.Append kvl_tabledef.CreateField("phone", dbText)
    On Error Resume Next
    CurrentDb.TableDefs.Append kvl_tabledef 'ajoute la table à la base
    kvl_recordset.Open "stefan_ram_table", CurrentProject.Connection, adOpenKeyset,
    adLockOptimistic
    kvl_recordset.AddNew
    kvl_recordset.Fields("name").value = "Peter"
    kvl_recordset.Fields("phone").value = "123456"
    kvl_recordset.Update
    kvl_recordset.AddNew
    kvl_recordset.Fields("name").value = "Mary"
    kvl_recordset.Fields("phone").value = "987654"
    kvl_recordset.Update
    kvl_recordset.Close
    Set kvl_recordset = Nothing
    kvl_xml = FreeFile()
    Open "stefan_ram.xml" For Output As kvl_xml
    Print #kvl_xml, "<table>"
    kvl_recordset.Open "stefan_ram_table", CurrentProject.Connection, adOpenKeyset,
    adLockOptimistic
    Do While Not kvl_recordset.EOF
        Print #kvl_xml, " <record>"
        Print #kvl_xml, " <name>" & kvl_recordset.Fields("name") & "</name>"
        Print #kvl_xml, " <phone>" & kvl_recordset.Fields("phone") & "</phone>"
        Print #kvl_xml, " </record>"
        kvl_recordset.MoveNext
    Loop
    Print #kvl_xml, "</table>"
    Close kvl_xml
```

```
        kvl_recordset.Close
        Set kvl_recordset = Nothing
End Sub
```

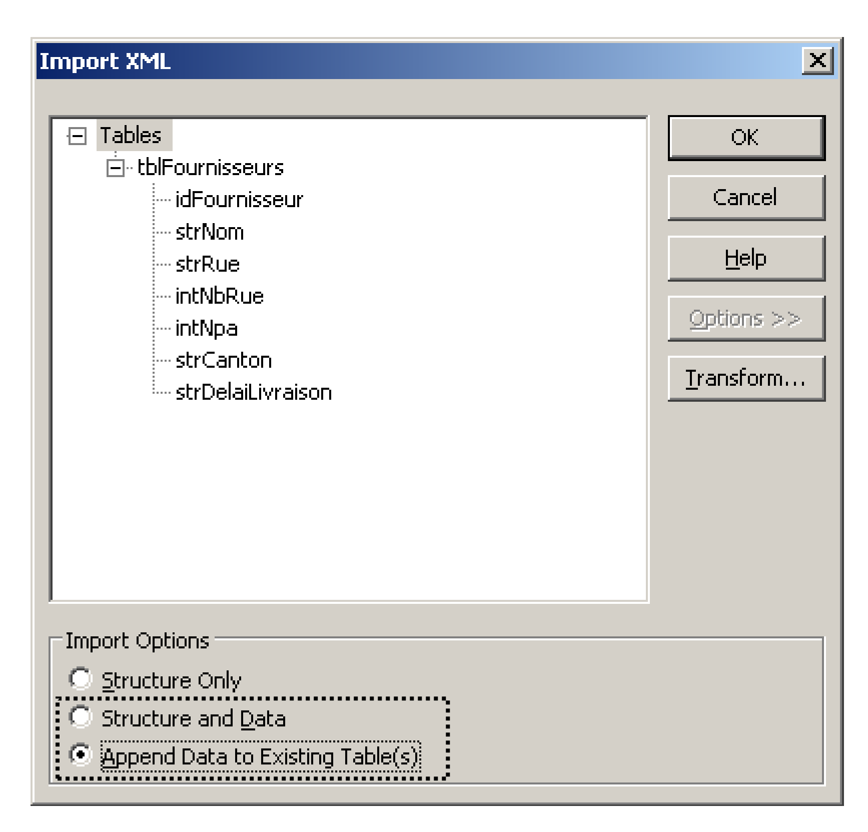
Inversement, il est possible d'importer une ficher XML par le code suivant (le nom de la table sera celui du nom du fichier XML):

```
Sub ImportTable()
   Application.ImportXML "C:\tblArticles.xml"
End Sub
```

C'est beau non….

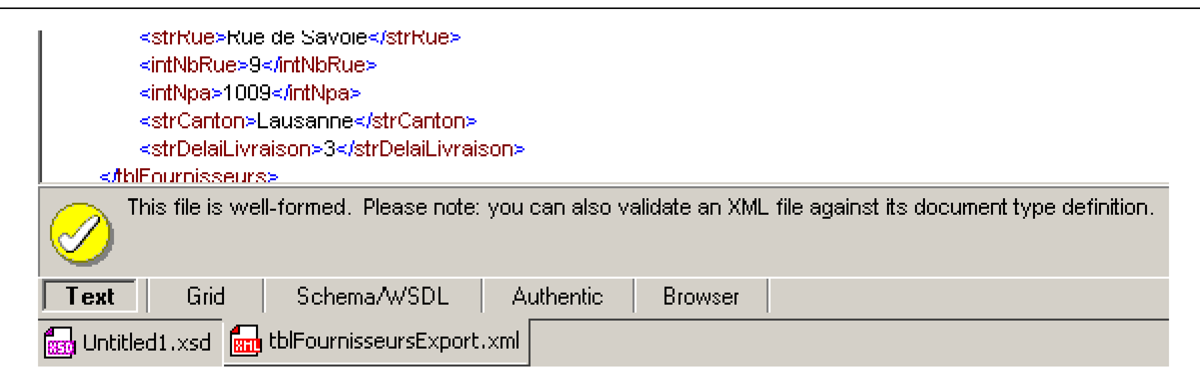
22.1 Import XML

Soit le fichier *tblFournisseursImport.xml* que votre formateur vous met à disposition.
Importez en les données dans la table *tblFournisseurs* existante en utilisant l'assistant manuel:



Exportez maintenant toute la table *tblFournisseur* au format XML sous le nom *tblFournisseurExport*.

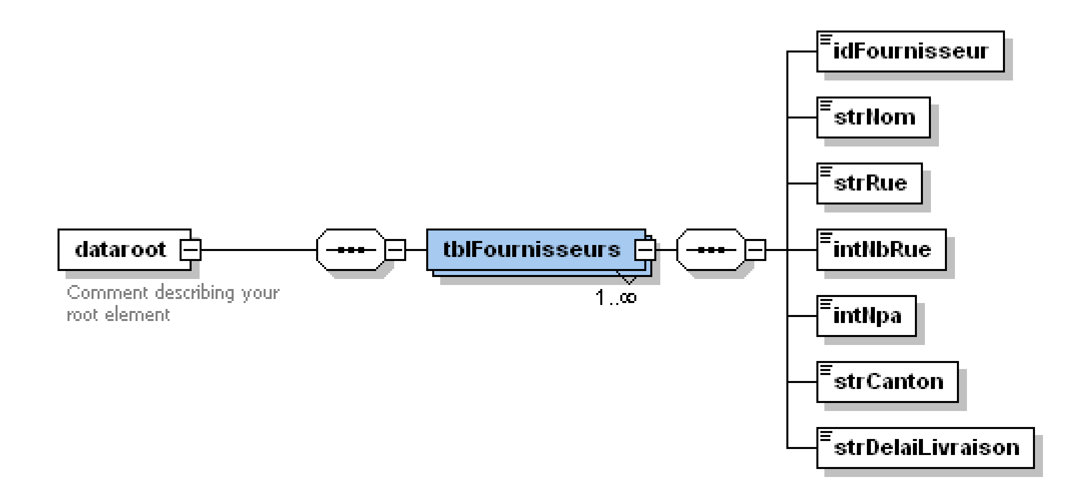
Si nous ouvrons ce fichier XML sortant dans un logiciel spécialisé comme XMLSpy, nous voyons bien que le fichier est conforme:



Pour créer un validateur (car en XML un fichier conforme et un fichier valide sont deux choses différentes), voici le fichier validateur XSD (obligatoire pour travailler dans MS Excel et MS Word 2003):

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="dataroot">
        <xs:annotation>
            <xs:documentation>Comment describing your root element</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="tblFournisseurs" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="idFournisseur" type="xs:integer"/>
                            <xs:element name="strNom" type="xs:string"/>
                            <xs:element name="strRue" type="xs:string"/>
                            <xs:element name="intNbRue" type="xs:integer"/>
                            <xs:element name="intNpa" type="xs:integer"/>
                            <xs:element name="strCanton" type="xs:string"/>
                            <xs:element name="strDelaiLivraison" type="xs:integer"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

et son schéma XMLSpy correspondant:



Il existe maintenant une infinité de possibilité pour mettre en forme notre fichier XML dans MS Word ou Internet Explorer à l'aide d'une feuille de mise en page XSL tel que:

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
<html>
<body>
    <table border="1" cellspacing="0" cellpadding="3">
        <tr bgcolor="#FFFF00">
            <td>Nom Fournisseur</td>
            <td>Adresse </td>
        </tr>
        <xsl:for-each select="dataroot/tblFournisseurs">
        <xsl:choose>
        <xsl:when test=".[strNom='Infolearn SA']">
        <tr bgcolor="#00FF00">
            <td><xsl:value-of select="strNom"/></td>
            <td><xsl:value-of select="strRue"/>nbsp<xsl:value-of select="intNbRue"/>nbsp<xsl:value-of select="intNpa"/>nbsp
<xsl:value-of select="strCanton"/></td>
        </tr>
        </xsl:when>
        <xsl:otherwise>
        <tr>
            <td><xsl:value-of select="strNom"/></td>
            <td><xsl:value-of select="strRue"/><xsl:value-of select="intNbRue"/><xsl:value-of select="intNpa"/><xsl:value-of
select="strCanton"/></td>
        </tr>
        </xsl:otherwise>
        </xsl:choose>
        </xsl:for-each>
    </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Appliqué au fichier XML, cela devient dans Internet Explorer:

Nom Fournisseur	Adresse
Duplibureau	Ruelle du Soleil 0 2003Neuchâtel
La maison du papier	Rue des Anges 17 1010Lausanne
Tartanpion	Grand-Rue 115 1205Genève
Infolearn SA	Rue de Savoie 9 1009 Lausanne
Herdt AG	Limatt Strasse 116 3234Zürich

Voilà ce sera tout pour notre découverte du monde merveilleux de l'XML.

Bien sûr, XML+Access+Word+Excel+InfoPath 2003 forment à eux 5 dans les outils classiques de la suite professionnelle MS Office un outil d'une puissance, et d'une compliance sans égal !

23 Reverse Engineering (rétro-conception)

Le but de cette partie du cours est d'apprendre à utiliser MS Visio en tant qu'outil de modélisation.

Remarque: Avoir suivi un cours MS Visio initiation au préalable est conseillé.

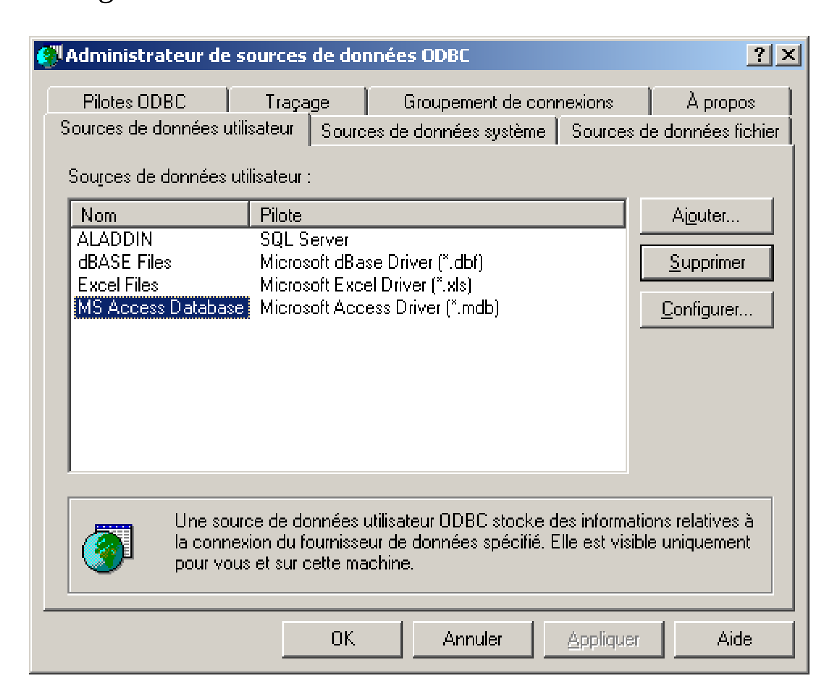
Nous avons déjà vu, au tout début de cet ouvrage, comment envoyer un modèle logique de données (MLD) de MS Visio à un modèle physique de données (MPD) MS Access.Quand une base de données a été faite sans modélisation ou qu'il faut refaire cette dernière (comme c'est souvent le cas à cause du manque de connaissance des amateurs ou plus simplement à cause de l'évolution des technologies), il est aussi important d'aller savoir importer le MPD d'une base existante dans un MLD MS Visio.

Dans un premier temps, nous allons importer la base de données que nous avons créée pendant ce cours. Pour ce faire, voici une méthode parmi tant d'autre.

1. Ouvrir le Panneau de configuration et ouvrir l'application permettant de gérer les sources de données ODBC (cet outil se trouve habituellement dans les outils d'administration et nécessite une version "Pro" de MS Windows).

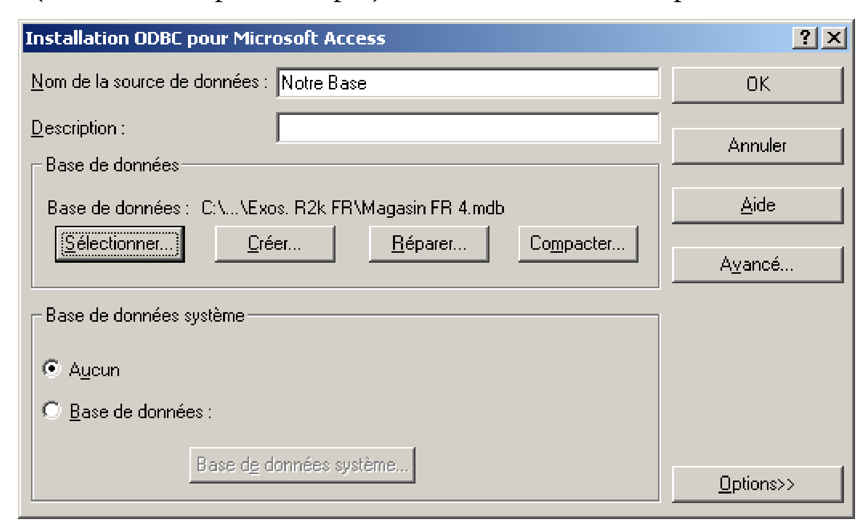


2. Aller dans l'onglet "Source de données utilisateur":



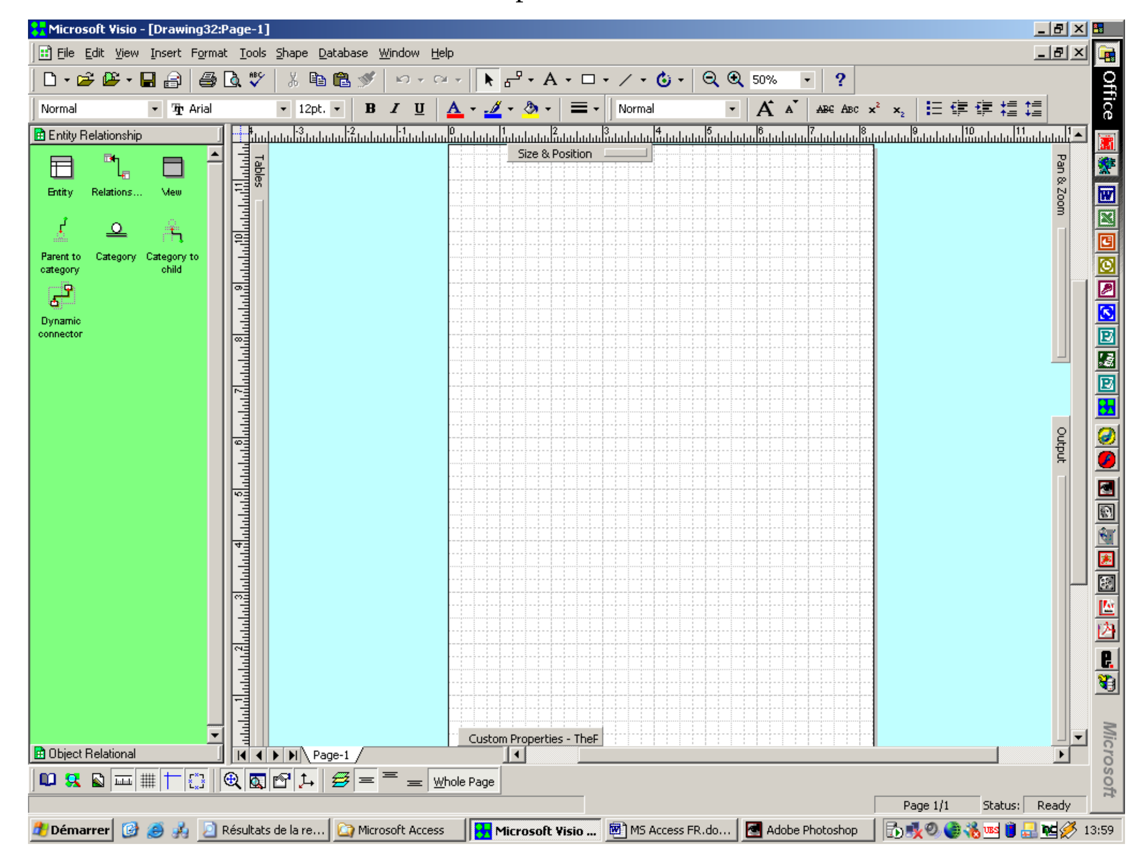
3. Cliquer sur ajouter et choisir un pilote ("driver" en anglais) de type MS Access et cliquer sur suivant.

4. Sélectionner la base de données qui a été créée lors des cours précédent et lui définir un nom ("Notre base" par exemple) et en donner un descriptif.

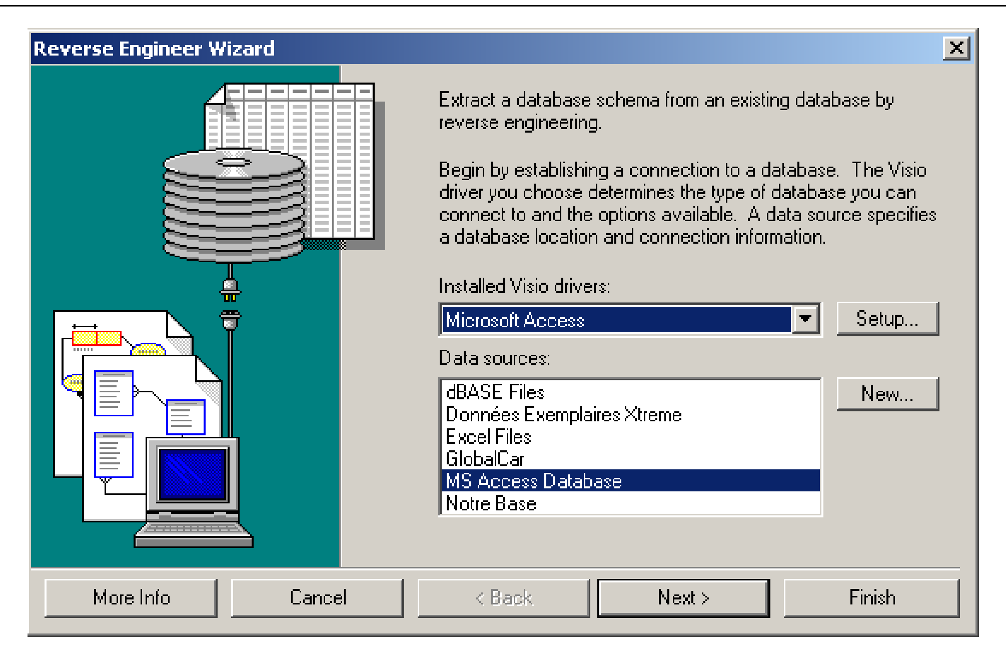


5. Cliquer sur le bouton "OK" et fermer l'application des sources de données ODBC et ouvris MS Visio

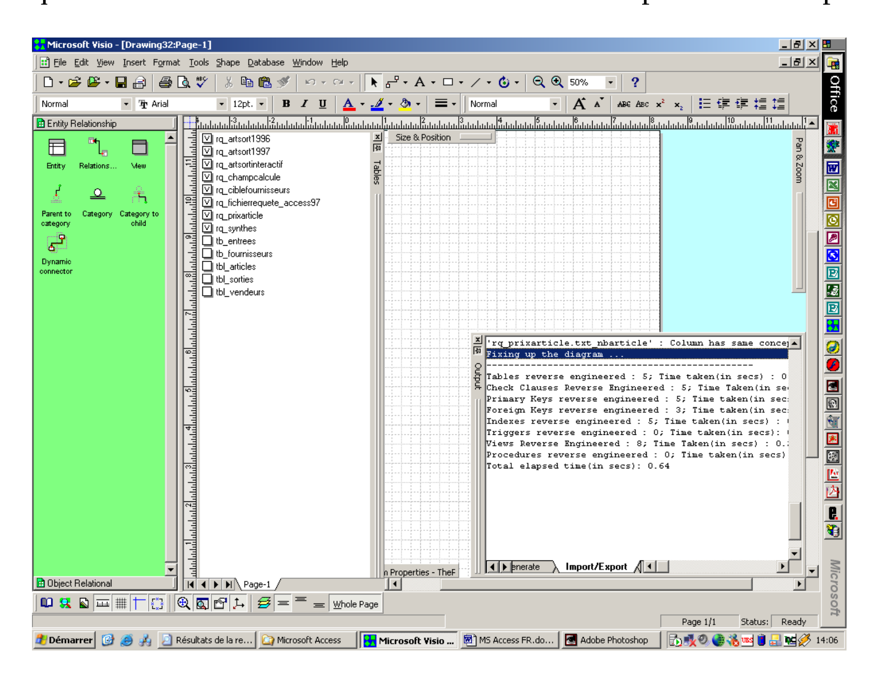
6. Dans MS Visio 2000 version Entreprise ou MS Visio 2002/2003 version Architecte créer un nouveau document vide à partir du modèle de base de données.



7. Aller dans le menu "Database" et sélectionner l'option "Reverse Engineer". Doit alors apparaître la boîte de dialogique ci-dessous:



8. Effectuer les mêmes choix de drivers et de données sources que ce qui est présenté sur la figure ci-dessus et cliquer sur "Next".
9. MS Visio va vous demander le nom d'utilisateur et le mot de passe d'accès à la base de données. S'il n'y en a pas, cliquer simplement sur le bouton "OK".
10. MS Visio vous demande quels sont les types d'éléments que vous souhaiteriez pouvoir importer. Prenez tout dans un premier temps et cliquez à nouveau sur "Next".
11. MS Visio vous demande maintenant parmi les types d'éléments choisis précédemment, quels sont les objets que vous souhaiteriez importer dans MS Visio. Sélectionnez tout (bouton: "Select All") et cliquez sur le bouton "Next".
12. et… cliquez sur le bouton "Finish"
13. Voilà à quoi devrait ressembler MS Visio une fois les opérations d'import effectuées:



24 A.S.P. et MDB

Pour se connecter à une base de données en ASP 3.0 (le langage lui-même fait l'objet d'un cours de plusieurs semaines) il faut créer de nouveaux objets:

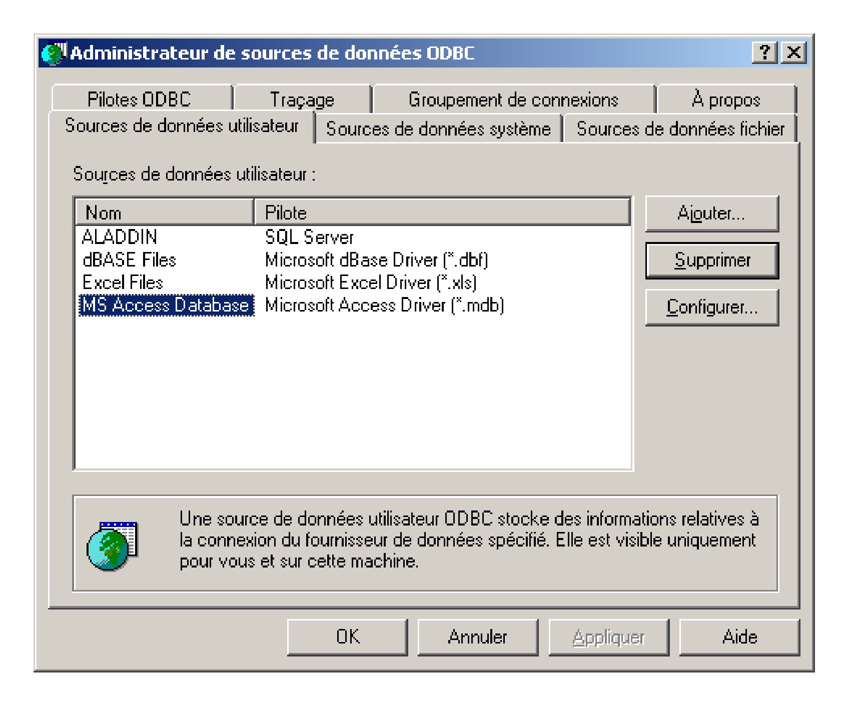
```
<%

Dim objConn
Set ObjConn = Server.CreateObject("AdoDB.Connection")
….
```

La méthode la plus souple pour se connecter à une base de données quelconque est de créer un D.S.N. (Data Source Name) en passant également par l'application:

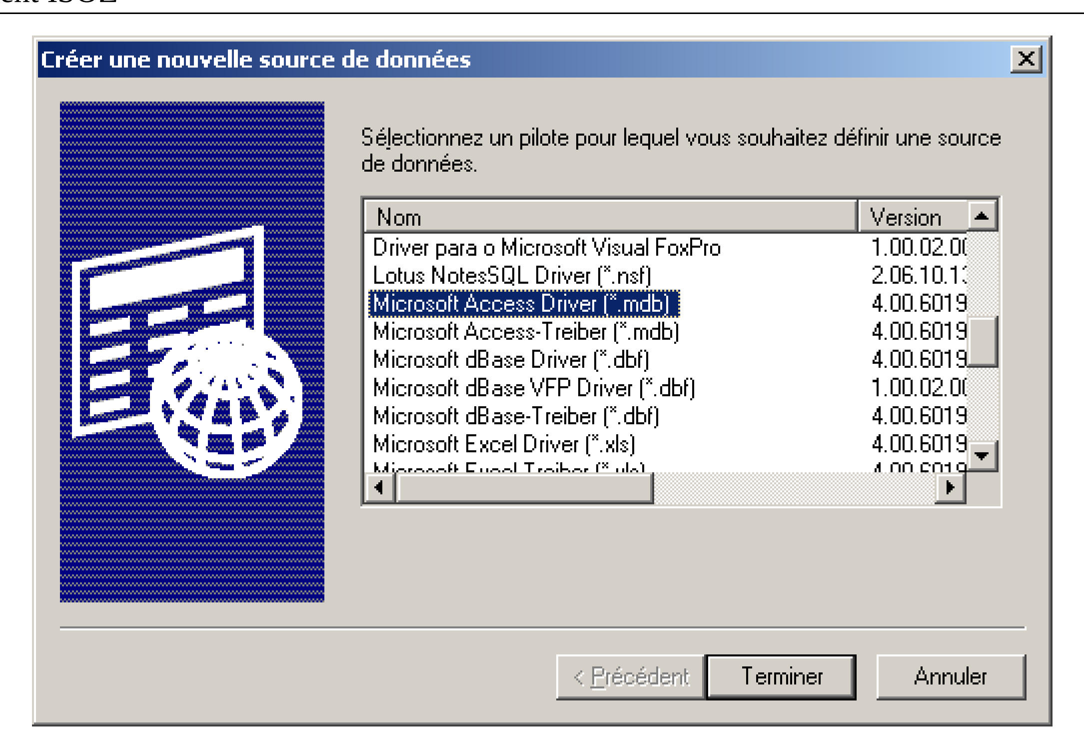


et par la fenêtre:

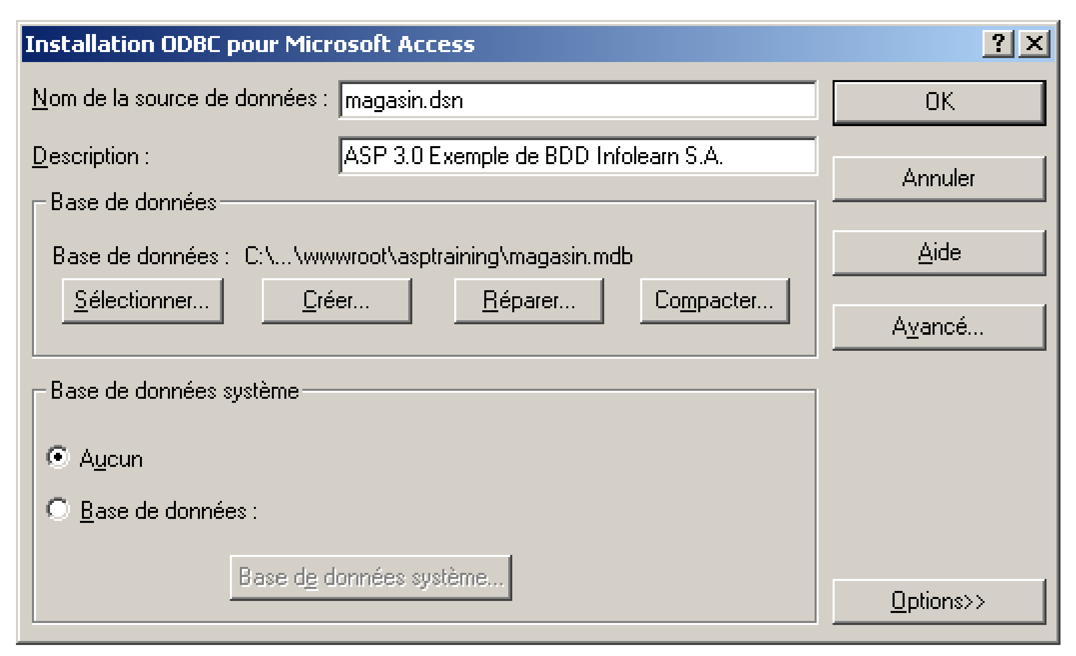


Mais avant de continuer, il faut placer la base MS Access dans un répertoire donné du wwwroot du serveur.

Comme il s'agit de créer une nouvelle source de données, cliquez sur le bouton *Ajouter*. Une liste de pilotes apparaît comme pour le chapitre précédent, à la différence qu'il faudra nommer la connexion "magasin.dsn" comme ci-dessous:



Cliquez sur *Terminer* et ensuite:



… et cliquez sur *OK*. Dans le liste des *Sources de données utilisateur*, il y devra y avoir *magasin.dsn*.

Nous disposons maintenant d'une source de données système. L'appel à l'objet *Connection* s'effectue ainsi:

```
<%
Dim objConn
Set ObjConn = Server.CreateObject("AdoDB.Connection")
objConn.ConnectionString = "DSN=magasin.dsn"
….
```

Il existe une alternative à la source de données système. Au lieu de placer les informations de connexion dans cette source, vous pouvez les placer dans une chaîne de connexion.

```
<%
Dim objConn
Set objConn = Server.CreateObject("ADODB.Connection")
objConn.ConnectionString="DRIVER={Microsoft Access Driver (*.mdb)};" &
"DBQ=magasin.mdb"
```

24.1 Affichage de données

Voici un exemple de code se connectant à la base et affichant le contenu simple de la table *tbl_articles*:

```
<%@LANGUAGE="VBSCRIPT"%>
<html>
<body>
  <%
    Dim objConn
    Set objConn = Server.CreateObject("ADODB.Connection")
    objConn.ConnectionString="DRIVER={Microsoft Access Driver (*.mdb)};" &
"DBQ=C:\magasin.mdb"
   objConn.Open
   'Crée une instance d'objet Recordset et extrait cette information
   'de la table tbl_articles.
   Dim objRS
   Set objRS = Server.CreateObject("ADODB.Recordset")
   objRS.Open "tbl_articles", objConn, , , adCmdTable
   'Affiche le contenu de la table Amis
   Do While Not objRS.EOF
       Response.Write "<B> Nom de l'article: " & objRS("txt_nbarticle") & "</B><BR>"
       Response.Write "Déscription: " & objRS("mem_designation") & "<BR>"
       Response.Write "Numéro de fournisseur: " & objRS("int_nbfournisseur") & ", " &
objRS("int_moq")
       Response.Write "<BR> Prix à l'unité: " & objRS("cur_unitprice") & "<BR><hr>"
       'Déplacement vers la prochaine rangée de la table Amis
       objRS.MoveNext
   Loop
   'Elimination des objets ADO
   objRS.Close
   Set objRS = Nothing
   objConn.Close
   Set objConn = Nothing
 %>
</body>
</html>
```

24.2 Ajout de données

Voici un exemple de code se connectant à la base, ajoutant un enregistrement et affichant le contenu de la table *tbl_articles* dans un tableau HTML:

```
<% @Language=VBScript %>
<%  Option Explicit %>
<%
'Attention l'exécution de cet exemple ne marche qu'une unique fois étant donné la clé
primaire
'dans la table access sur txt_nbarticles
Dim objConn
Set objConn = Server.CreateObject("ADODB.Connection")
objConn.ConnectionString="DRIVER={Microsoft Access Driver (*.mdb)};" &
"DBQ=C:\Inetpub\wwwroot\asptraining\magasin.mdb"
objConn.Open
'Crée une instance d'objet Recordset et extrait cette information
'de la table Amis.
Dim objRS
Set objRS = Server.CreateObject("ADODB.Recordset")
Const adCmdTable=2
Const adLockOptimistic=3
objRS.Open "tbl_articles", objConn, , adLockOptimistic, adCmdTable
%>
<HTML>
  <BODY>
<%
 objRS.AddNew
        objRS("txt_nbarticle") = "INF-005"
        objRS("mem_designation") = "Livre ASP3"
        objRS("int_nbfournisseur") = "2"
        objRS("int_moq") = "4"
        objRS("cur_unitprice") = "70"
 objRS.Update
 objRS.MoveFirst
%>
 <B>Entrées dans la table</B>
 <P>
 <TABLE>
 <TR>
    <TD>Article</TD>
    <TD>Désignation</TD>
    <TD>Numéro fournisseur</TD>
        <TD>M.O.Q.</TD>
        <TD>Prix à l'unité</TD>
 </TR>
<% Do While Not objRS.EOF %>
   <TR>
    <TD><%=objRS("txt_nbarticle") %></TD>
```

```
        <TD><%=objRS("mem_designation") %></TD>
        <TD><%=objRS("int_moq") %></TD>
        <TD><%=objRS("cur_unitprice") %></TD>
      </TR>
  <%
      objRS.MoveNext
    Loop
    objRS.Close
    Set objRS = Nothing
    objConn.Close
    Set objConn = Nothing
  %>
    </TABLE>
    </BODY>
  </HTML>
```

24.3 Mise à jour d'enregistrements

Et pour mettre à jour des enregistrements déjà existants:

```
<% @Language=VBScript %>
   <%  Option Explicit %>
   <%
   Dim objConn
   Set objConn = Server.CreateObject("ADODB.Connection")
   objConn.ConnectionString="DRIVER={Microsoft Access Driver (*.mdb)};" &
"DBQ=C:\Inetpub\wwwroot\asptraining\magasin.mdb"
   objConn.Open
   Dim objRS
   Set objRS = Server.CreateObject("ADODB.Recordset")
   Const adCmdTable=2
   Const adLockOptimistic=3
   objRS.Open "tbl_articles", objConn, , adLockOptimistic, adCmdTable
   %>
   <HTML>
     <BODY>
    <B>Entrées dans la table AVANT mise à jour</B>
    <P>
    <TABLE>
    <TR>
        <TD>Article</TD>
        <TD>Désignation</TD>
        <TD>Numéro fournisseur</TD>
                <TD>M.O.Q.</TD>
                <TD>Prix à l'unité</TD>
    </TR>
  <% Do While Not objRS.EOF %>
      <TR>
        <TD><%=objRS("txt_nbarticle") %></TD>
        <TD><%=objRS("mem_designation") %></TD>
```

```
        <TD><%=objRS("int_nbfournisseur") %></TD>
        <TD><%=objRS("int_moq") %></TD>
        <TD><%=objRS("cur_unitprice") %></TD>
      </TR>
  <%
      objRS.MoveNext
    Loop
    objRS.MoveFirst
                    objRS("txt_nbarticle") = "gen007"
                    objRS("mem_designation") = "Livre ASP.Net"
                    objRS("int_nbfournisseur") = "5"
                    objRS("int_moq") = "5"
                    objRS("cur_unitprice") = "120"
    objRS.Update
    objRS.MoveFirst
  %>
    </TABLE>
    <P>
          <B>Entrées dans la table APRÈS mise à jour</B>
          <P>
    <TABLE>
  <% Do While Not objRS.EOF %>
      <TR>
        <TD><%=objRS("txt_nbarticle") %></TD>
        <TD><%=objRS("mem_designation") %></TD>
        <TD><%=objRS("int_nbfournisseur") %></TD>
        <TD><%=objRS("int_moq") %></TD>
        <TD><%=objRS("cur_unitprice") %></TD>
      </TR>
  <%
      objRS.MoveNext
    Loop
    objRS.Close
    Set objRS = Nothing
    objConn.Close
    Set objConn = Nothing
  %>
    </TABLE>
    </BODY>
  </HTML>
```

Et pour supprimer un enregistrement:

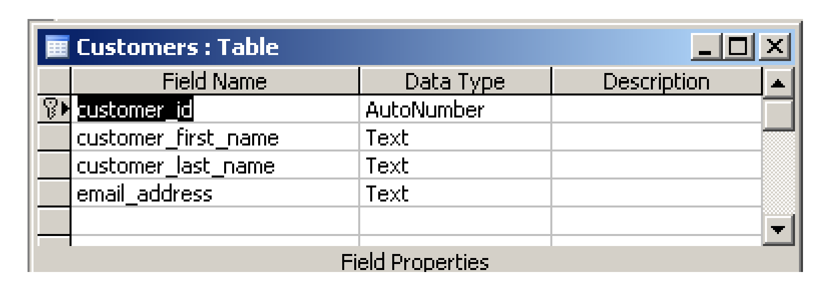
```
   <% @Language=VBScript %>
   <%  Option Explicit %>
   <%
   Dim objConn
   Set objConn = Server.CreateObject("ADODB.Connection")
   objConn.ConnectionString="DRIVER={Microsoft Access Driver (*.mdb)};" &
"DBQ=C:\Inetpub\wwwroot\asptraining\magasin.mdb"
```

```
 objConn.Open
 Dim objRS
 Set objRS = Server.CreateObject("ADODB.Recordset")
 Const adCmdTable=2
 Const adLockOptimistic=3
 objRS.Open "tbl_articles", objConn, , adLockOptimistic, adCmdTable
 %>
 <HTML>
  <BODY>
  <B>Entrées dans la AVANT LA SUPRESSION:</B>
  <P>
  <TABLE border="1">
  <TR>
     <TD>Article</TD>
     <TD>Désignation</TD>
     <TD>Numéro fournisseur</TD>
                  <TD>M.O.Q.</TD>
                  <TD>Prix à l'unité</TD>
  </TR>
<% Do While Not objRS.EOF %>
    <TR>
      <TD><%=objRS("txt_nbarticle") %></TD>
      <TD><%=objRS("mem_designation") %></TD>
      <TD><%=objRS("int_nbfournisseur") %></TD>
      <TD><%=objRS("int_moq") %></TD>
      <TD><%=objRS("cur_unitprice") %></TD>
    </TR>
<%
    objRS.MoveNext
  Loop
  objRS.MoveFirst
  objRS.Delete
  objRS.MoveFirst
%>
  </TABLE>
  <P>
        <B>Entrées dans la APRES LA SUPRESSION:</B>
  <P>
  <TABLE border="1">
<% Do While Not objRS.EOF %>
    <TR>
      <TD><%=objRS("txt_nbarticle") %></TD>
      <TD><%=objRS("mem_designation") %></TD>
      <TD><%=objRS("int_nbfournisseur") %></TD>
      <TD><%=objRS("int_moq") %></TD>
      <TD><%=objRS("cur_unitprice") %></TD>
    </TR>
<%
    objRS.MoveNext
  Loop
```

```
  objRS.Close
  Set objRS = Nothing
  objConn.Close
  Set objConn = Nothing
%>
  </TABLE>
  </BODY>
</HTML>
```

25 P.H.P. et MDB

Considérons toujours la base de données *Magasin.mdb* avec la table suivante:



Field Name	Data Type	Description
customer_id	AutoNumber	
customer_first_name	Text	
customer_last_name	Text	
email_address	Text	

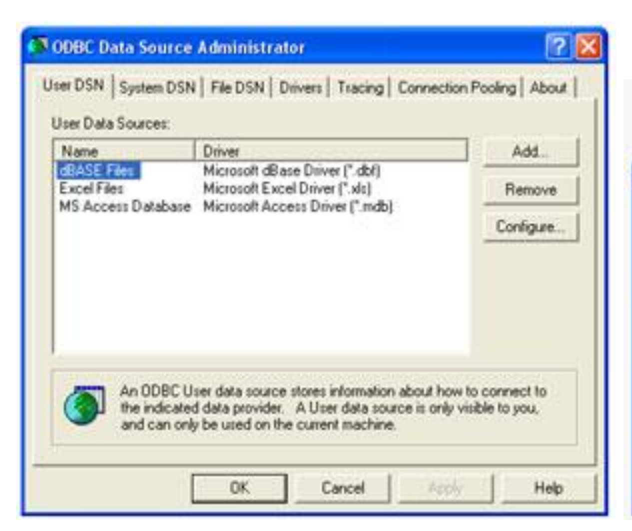
Contenant les données ci-dessous:

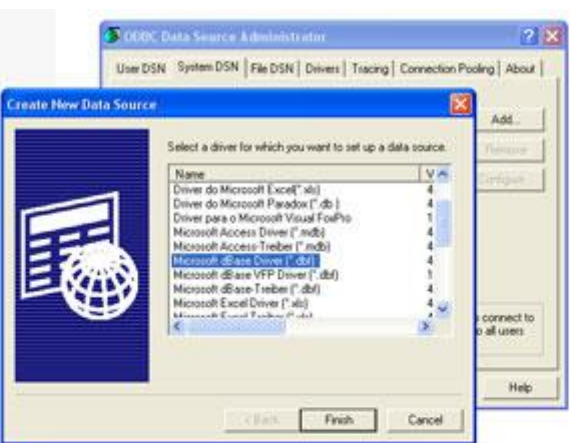


customer_id	customer_first_name	customer_last_name	email_address
1	John	Doe	john@doe.com

Créons ensuite un dossier nommé *phpaccess* dans notre sur notre serveur Apache (voir la procédure avec le formateur) dans lequel vous mettrez la base de données Magasin et dans lequel nous vous demandons de créer deux fichiers PHP vides pour l'instant: *odbc.php* et *query.php*.

Avant de continuer installons le pilote ODBC tel que nous l'avons fait pour l'exemple en ASP:







et validez le tout par OK. Ensuite, dans fichier *odbc.php* saisissez le code suivant (attention à la version du PHP !):

```
<?
$odbc = odbc_connect ('TestDB', 'root', '') or die( "Could Not Connect to ODBC Database!");

if (function_exists('odbc_fetch_array'))
return;

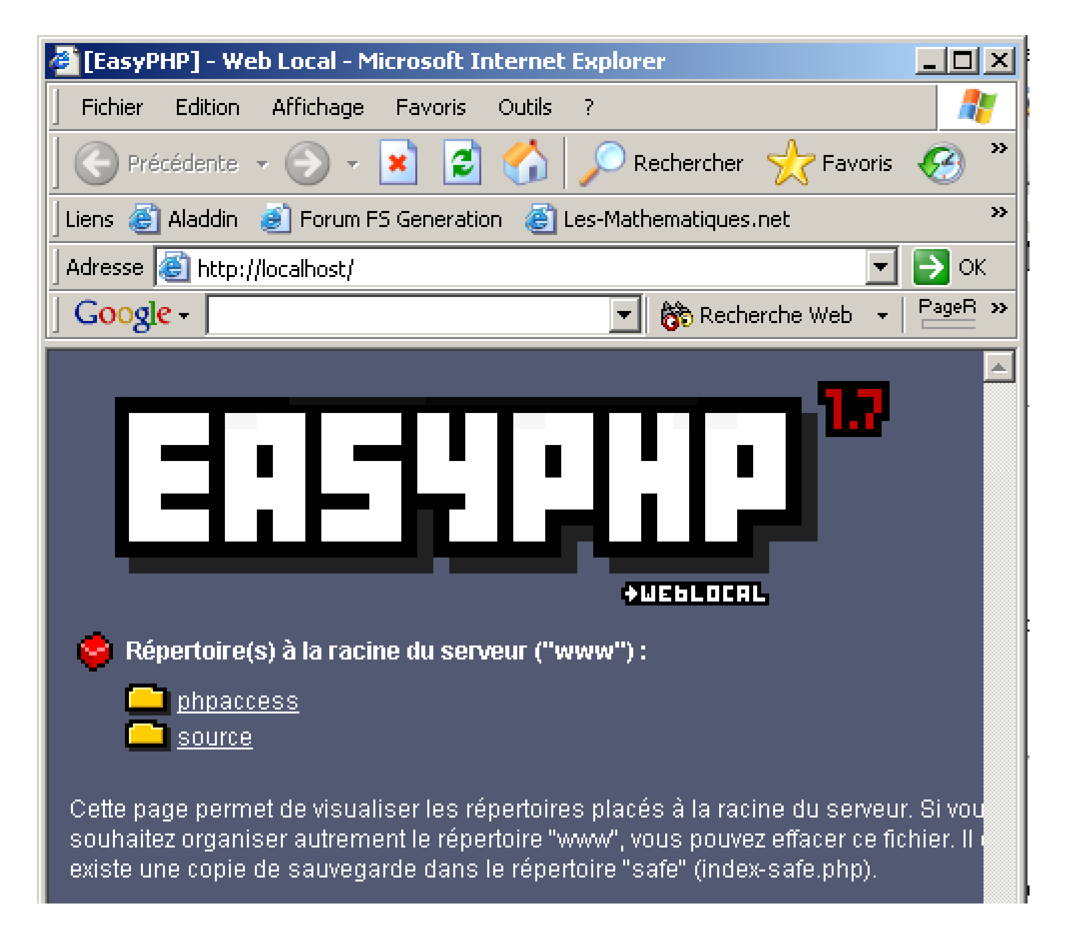
function odbc_fetch_array($result, $rownumber=-1) {
	if (PHP_VERSION > "4.1") {
		if ($rownumber < 0) {
			odbc_fetch_into($result, $rs);
		} else {
			odbc_fetch_into($result, $rs, $rownumber);
		}
	} else {
		odbc_fetch_into($result, $rownumber, $rs);
	}
	$rs_assoc = Array();
	foreach ($rs as $key => $value) {
		$rs_assoc[odbc_field_name($result, $key+1)] = $value;
	}
	return $rs_assoc;
}
?>
```

et dans le fichier *query.php*:

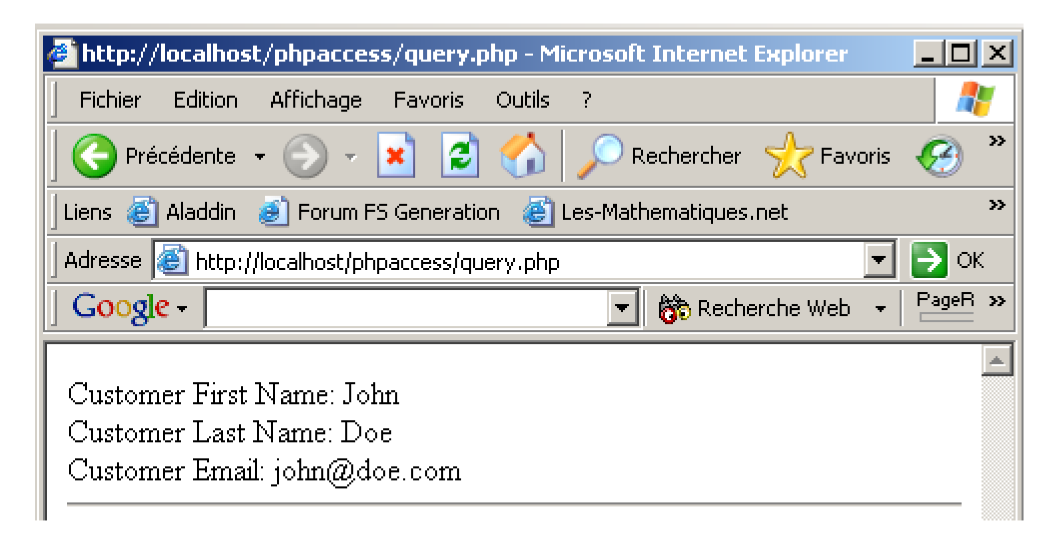
```
<?
include 'odbc.php';
$query = odbc_exec($odbc, "SELECT * FROM Customers") or die (odbc_errormsg());
while($row = odbc_fetch_array($query)){
	echo "Customer First Name: ".$row['customer_first_name']."<br />";
	echo "Customer Last Name: ".$row['customer_last_name']."<br />";
	echo "Customer Email: ".$row['email_address']."<br />";
	echo "<hr />";
```

```
}
odbc_close($odbc);
?>
```

Tapez ensuite http://localhost dans votre navigateur Internet (exemple effectué avec EasyPHP 1.7):



Cliquez sur le dossier *phpaccess* et ensuite sur *query.php*. Le résultat suivant devrait apparaître à l'écran:



26 MS Infopath

Infopath est un des trois nouveaux éléments de la suite MS Office System 2003 (InfoPath, OneNote, Business Contact Manager). Il met trivialement en évidence la stratégie de Microsoft de proposer des outils bureautiques performants de gestion et traitement de données au format XML.

MS Infopath est un logiciel de création de formulaires dont les possibilités sont infiniment plus grandes que MS Word et même Adobe Acrobat 6.0.

Nous allons ainsi faire deux exemples. Le premier sera quasi-statique simple ("plat") et utilisera des connaissances accessibles presque à n'importe qui. Le second exemple, complètement dynamique, demandera des connaissances de MS Access.

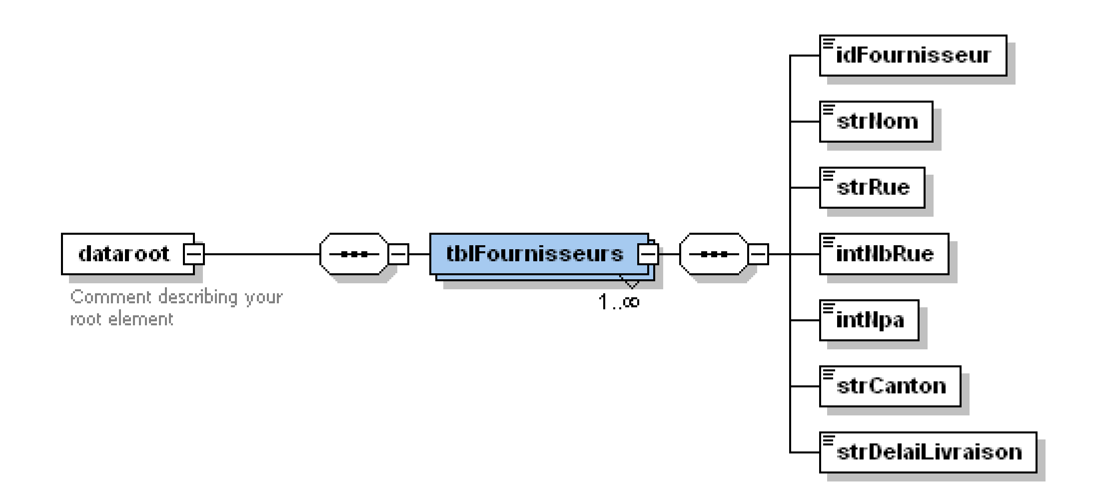
Remarque: l'utilisation ayant été grandement simplifiée depuis la sortie de son SP1 il est important de savoir que les exemples donnés ci-dessous ont été tirés des captures d'écran de la version MS InfoPath 2003 SP1.

26.1 Ajout de données

Commençons par l'exemple simple. Nous allons créer un formulaire InfoPath qui permet d'ajouter à notre base Access manuellement de nouveaux fournisseurs (nous faisons donc référence aux exemples donnés concernant MS Access 2003 précédemment).

Le but de ce document n'était pas d'apprendre MS InfoPath mais de voir le lien qu'il y a entre lui et le XML, je ne m'étendrai pas sur la manière de créer le formulaire en soi (c'est accessible même a quelqu'un qui débute en informatique).

Mais avant de créer le formulaire, il nous faut le validateur XSD de la table *tblFournisseurs* de notre base *.mdb. Rappelons qu'il est le suivante:



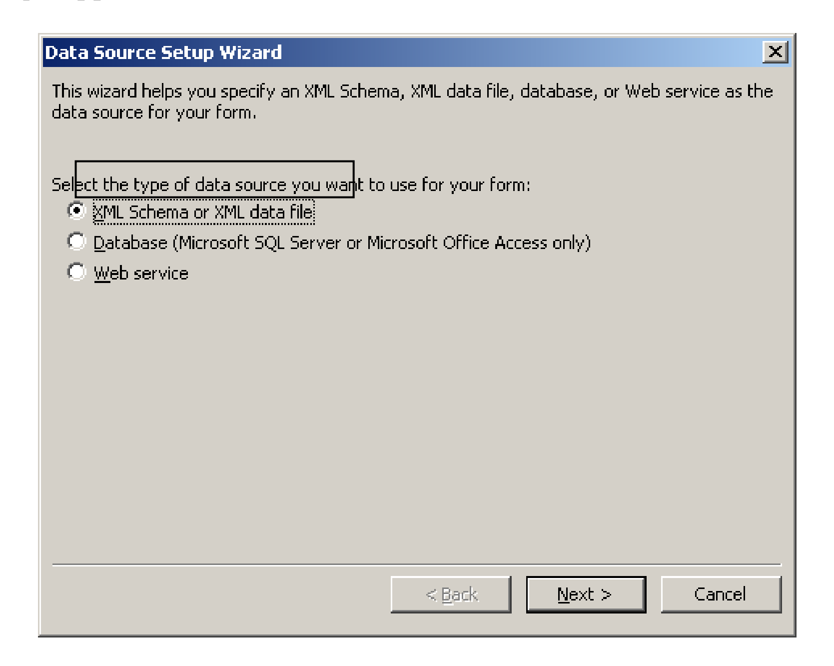
Maintenant, quand vous ouvrez MS InfoPath, dans le volet Office choisissez:



et ensuite toujours dans le volet Office:



Dans la fenêtre qui apparaît sélectionnez:

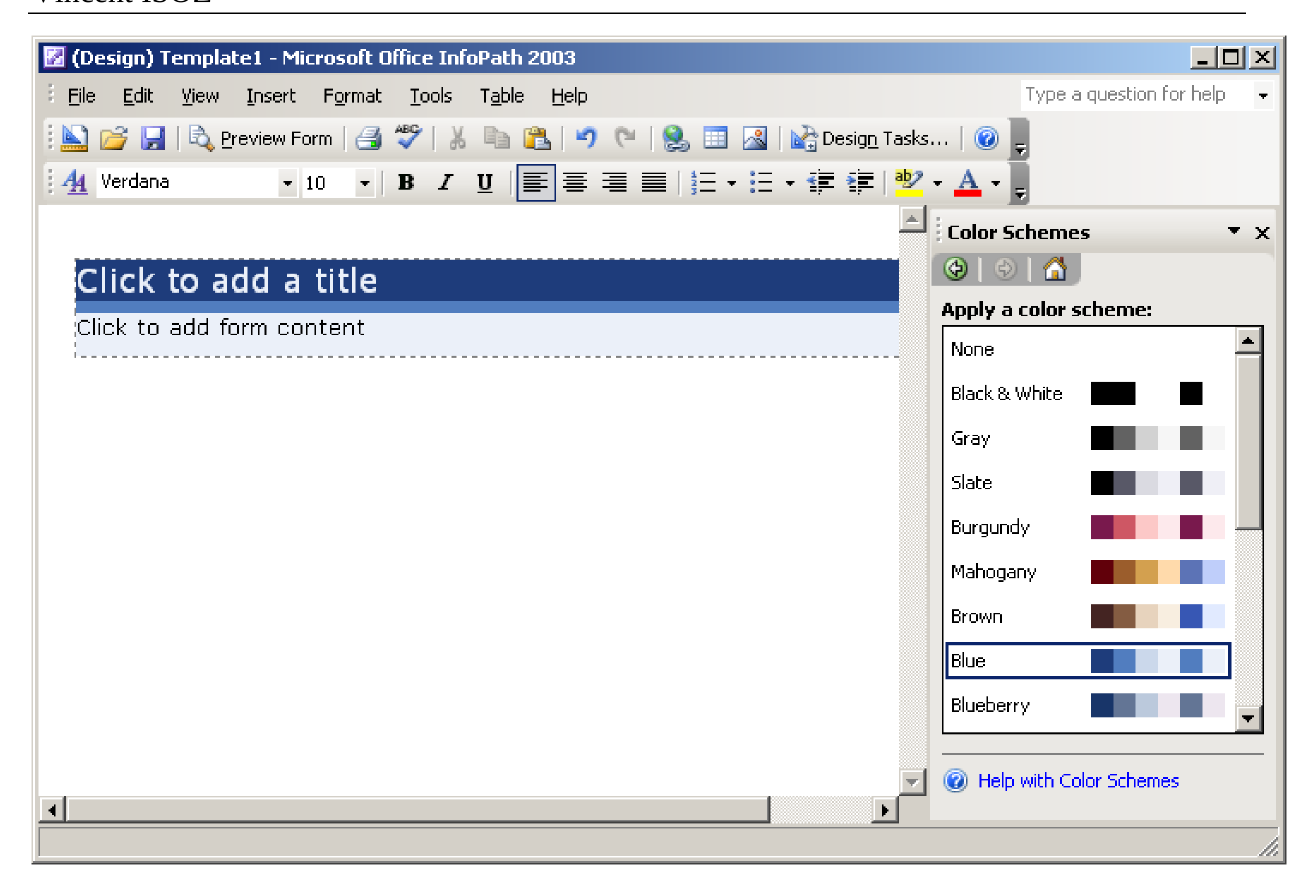


et allez chercher le fichier XSD. Une fois ceci fait vous devriez avoir dans le volet Office:

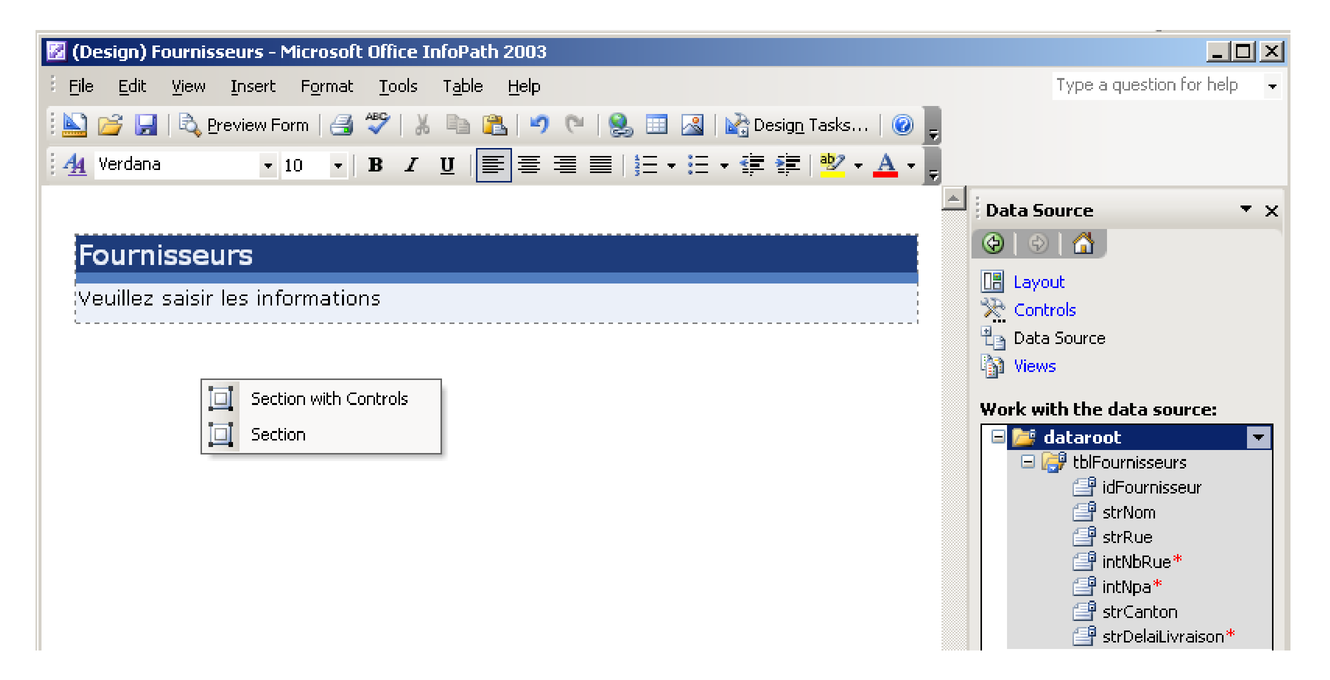


Il s'agit bien de la structure de notre XSD.

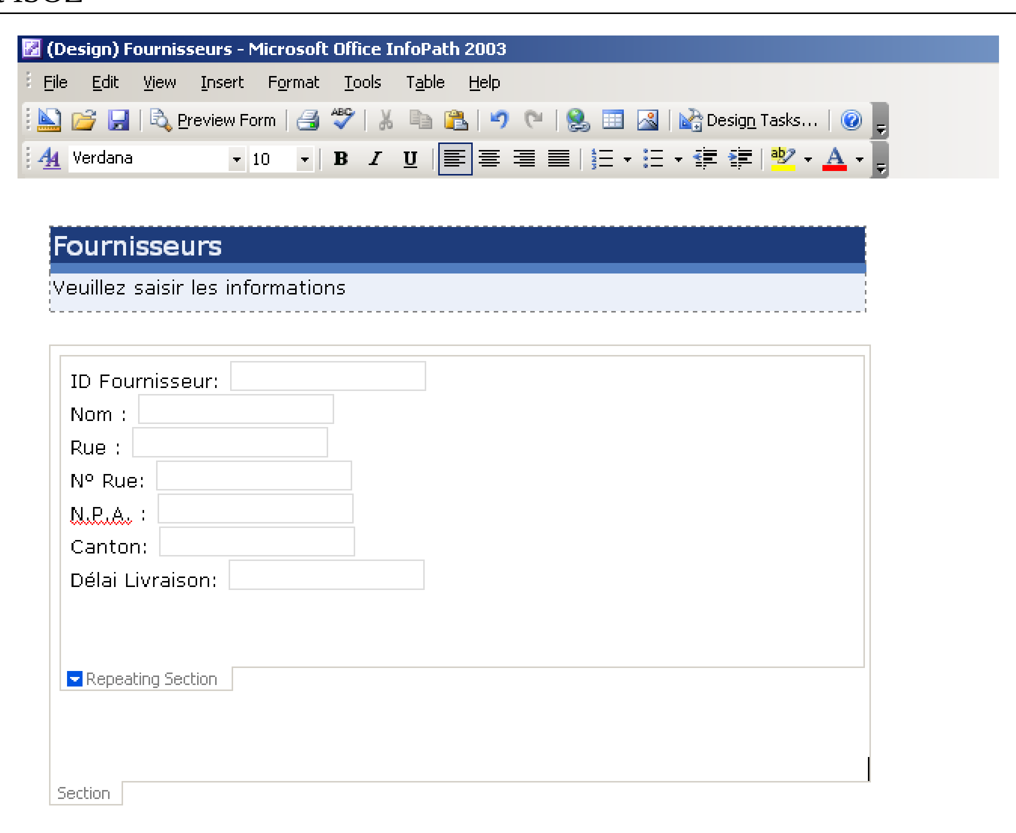
Ensuite, dans MS InfoPath choisissons un joli design pour commencer:



Ensuite retournez dans *Data Source* et faites un glisser-déplacer du nœud *dataroot* dans la feuille en sélectionnant *Section with controls*:



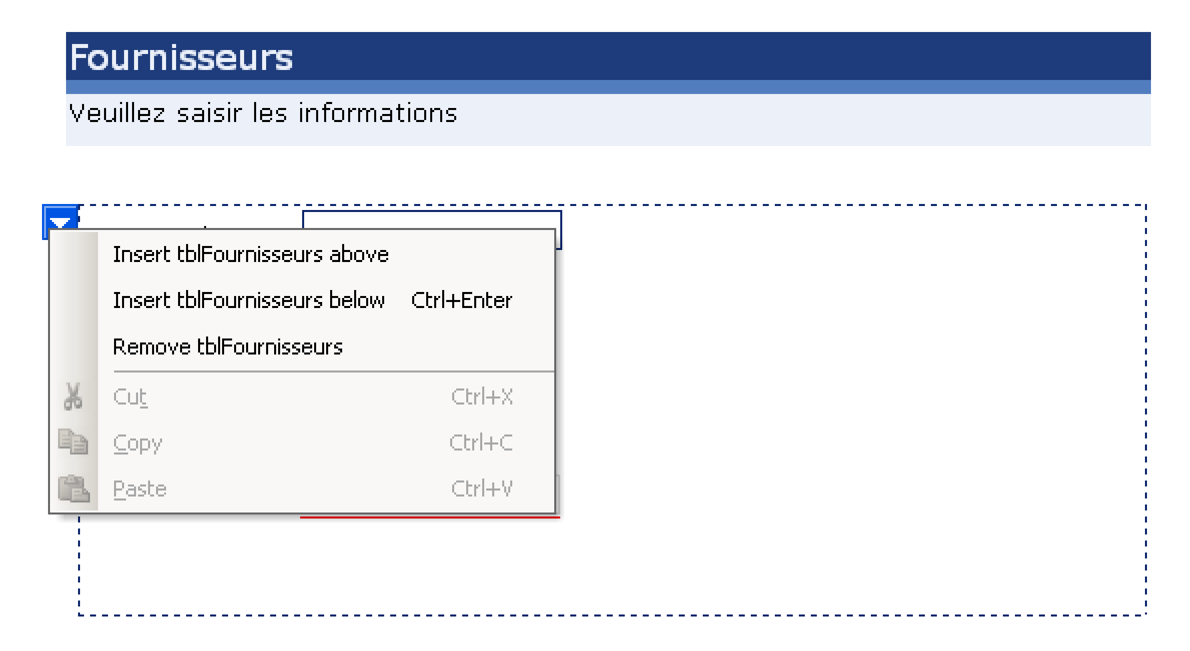
et voilà la résultat (après avoir un peu changé les texte des libellés):



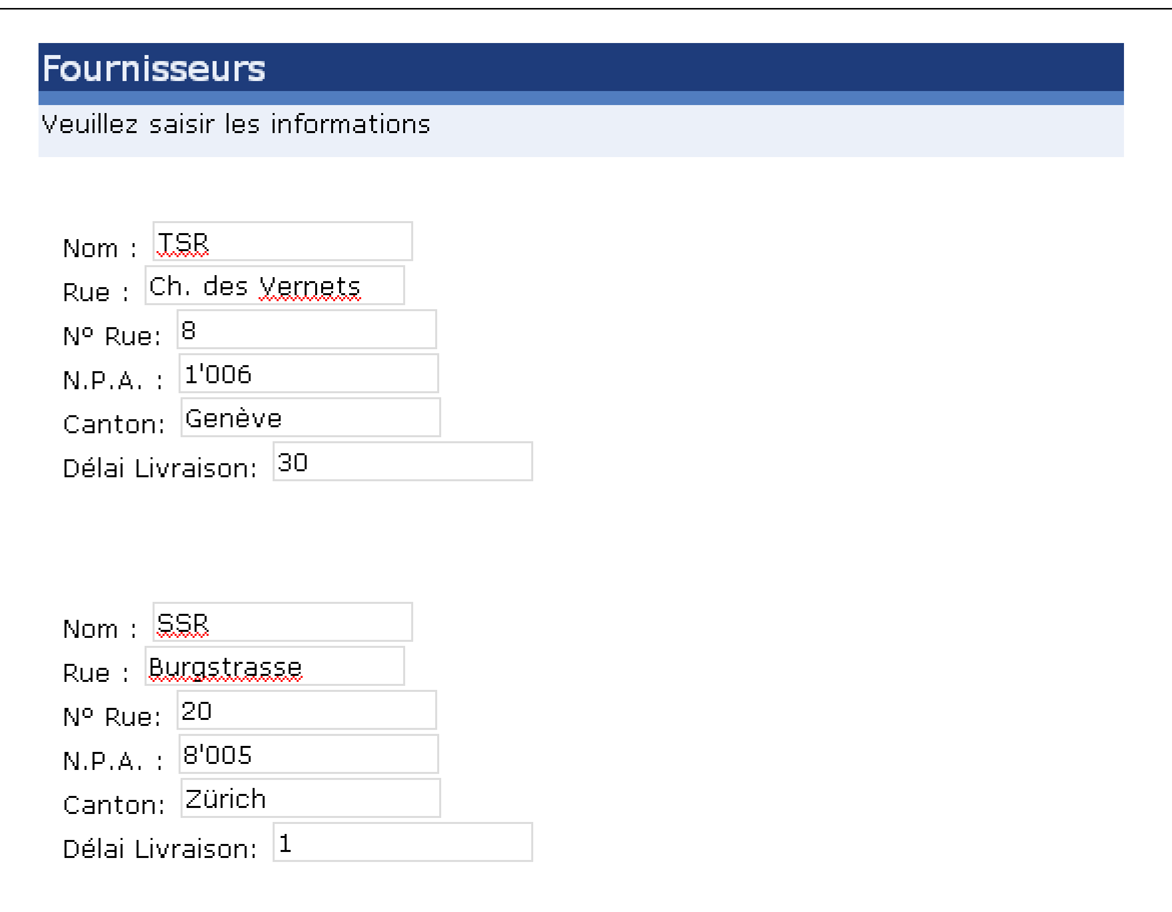
Enregistrez les modification et fermez InfoPath. Faites ensuite un double clique sur le fichier InfoPath:



et saisissez les données. Au besoin, en faisant cliquant sur la flèche bleue, vous pouvez ajouter des nouveaux enregistrements (c'est ce qui fait d'InfoPath un outil très puissant):



et donc après avoir saisi quelques valeurs:


Fournisseurs

Veuillez saisir les informations

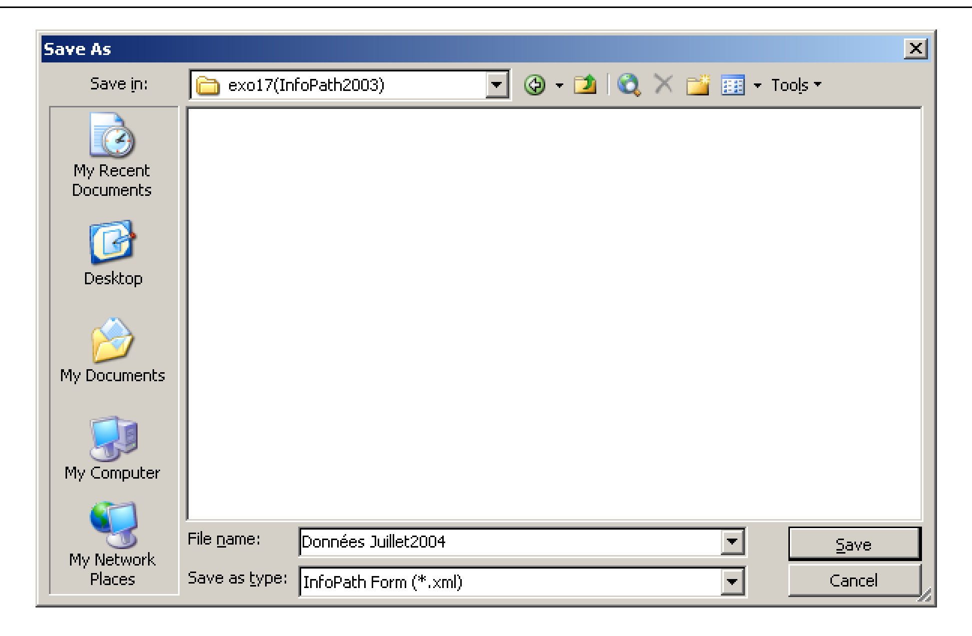
Nom :	TSR
Rue :	Ch. des Vernets
N° Rue:	8
N.P.A. :	1'006
Canton:	Genève
Délai Livraison:	30

Nom :	SSR
Rue :	Burgstrasse
N° Rue:	20
N.P.A. :	8'005
Canton:	Zürich
Délai Livraison:	1

L'utilisateur du formulaire clique sur *Enregistrer:*



et InfoPath propose automatiquement de sauvegarder les données en XML:



et voici le fichier XML sortant:

```
<?xml version="1.0" encoding="UTF-8"?>
<?mso-infoPathSolution solutionVersion="1.0.0.3" productVersion="11.0.5531" PIVersion="1.0.0.0"
href="file:///C:\Documents%20and%20Settings\Isoz\Mes%20documents\Professionel\Cours\XML\exo17(InfoPath2003)\Fournisseurs.xsn" language="fr" ?>
<?mso-application progid="InfoPath.Document"?>
<dataroot>
    <tblFournisseurs>
        <idFournisseur/>
        <strNom>TSR</strNom>
        <strRue>Ch. des vernets</strRue>
        <intNbRue>8</intNbRue>
        <intNpa>1006</intNpa>
        <strCanton>Genève</strCanton>
        <strDelaiLivraison>30</strDelaiLivraison>
    </tblFournisseurs>
    <tblFournisseurs>
        <idFournisseur/>
        <strNom>SSR</strNom>
        <strRue>Burgstrasse</strRue>
        <intNbRue>20</intNbRue>
        <intNpa>8005</intNpa>
        <strCanton>Zürich</strCanton>
        <strDelaiLivraison>1</strDelaiLivraison>
    </tblFournisseurs>
</dataroot>
```

Tout est parfait (mis à part la ligne propriétaire de Microsoft…) pour un traitement par MS Excel 2003, pour un formatage web par un XSL, pour une utilisation de pagination dans MS Word 2003 ou pour un import dans MS Access 2003. Rien de nouveau donc de ce côté.

L'employé n'a qu'a envoyer ce document XML par e-mail à son responsable ou à des collègues.

26.2 Echange de données

Nous allons maintenant faire un formulaire dynamique lié directement à la table *tblFournisseurs* de notre base MS Access avec des listes déroulantes, etc. etc. L'échange des données entre MS Access et MS InfoPaths se faisant par un flux XML.

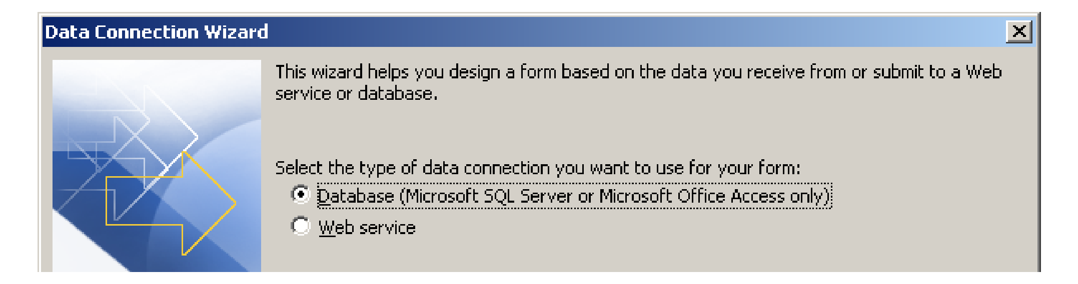
Ouvrez InfoPath (SP1) et sélectionnez:



Dans le volet Office cliquez sur:

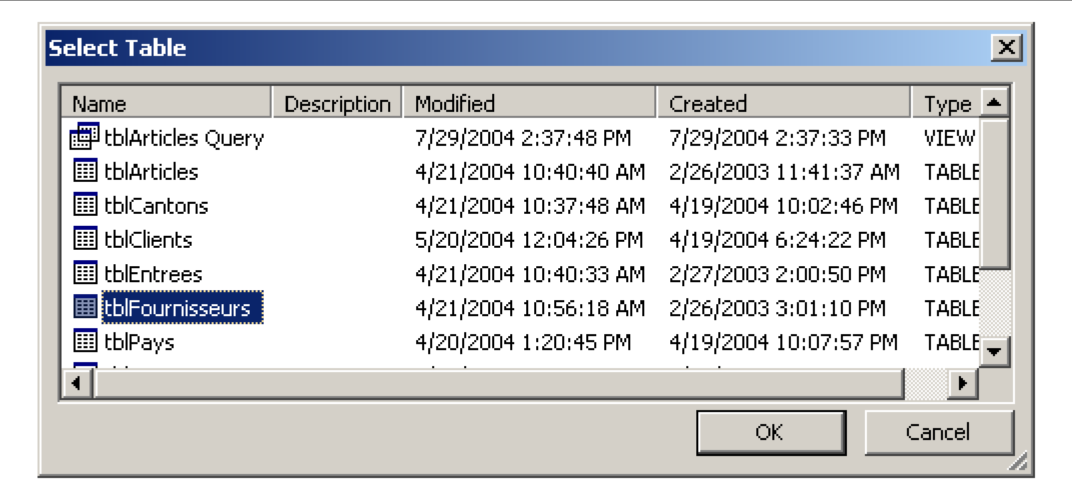


Afin que nous puissions nous connecter à la base MS Access:

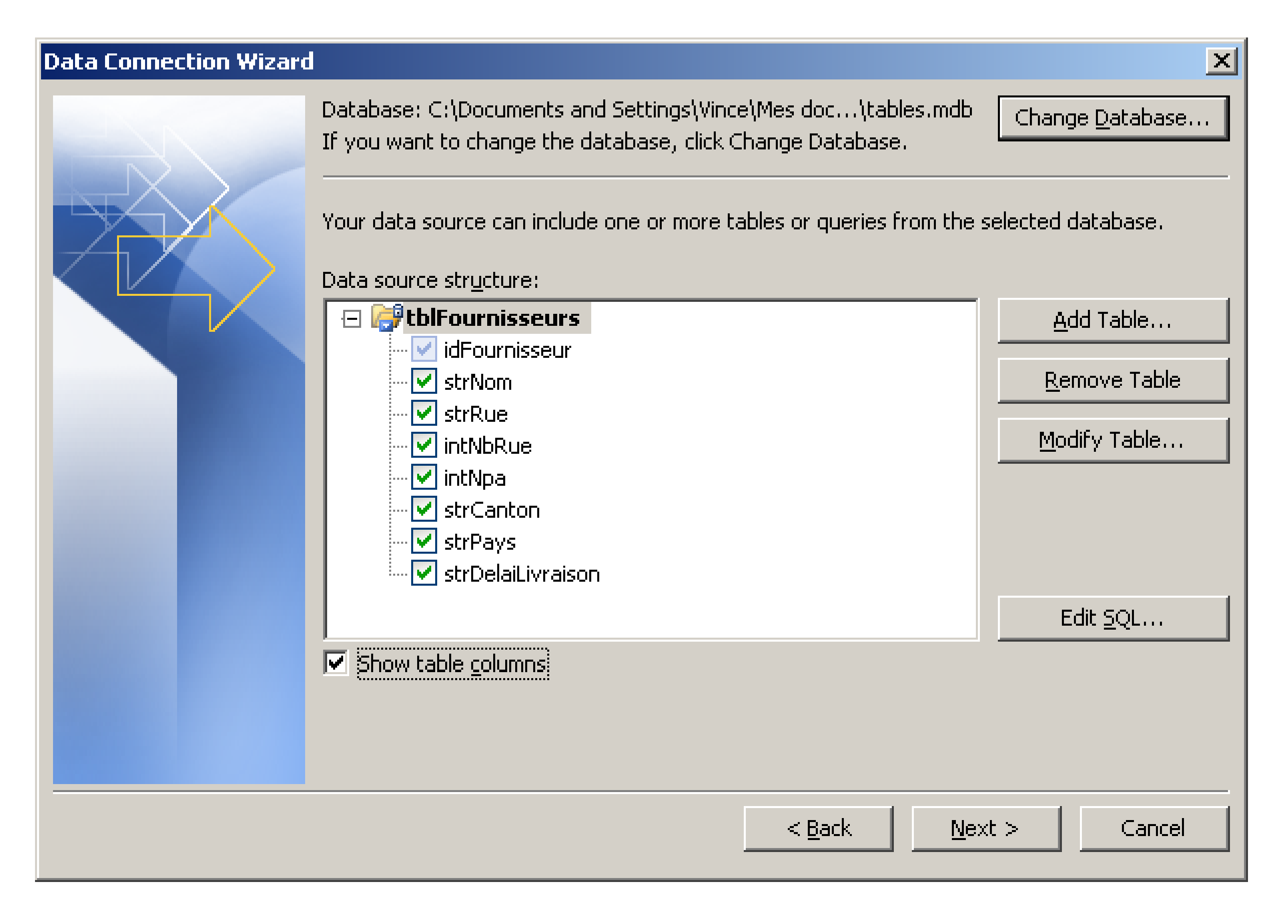


Ensuite la fenêtre suivante apparaît (cliquez sur *Next*):

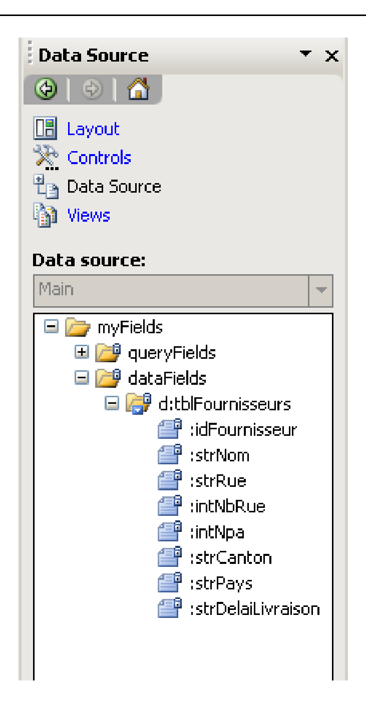
et après avoir sélectionné *tables.mdb* sélectionnez *tblFournisseurs:*



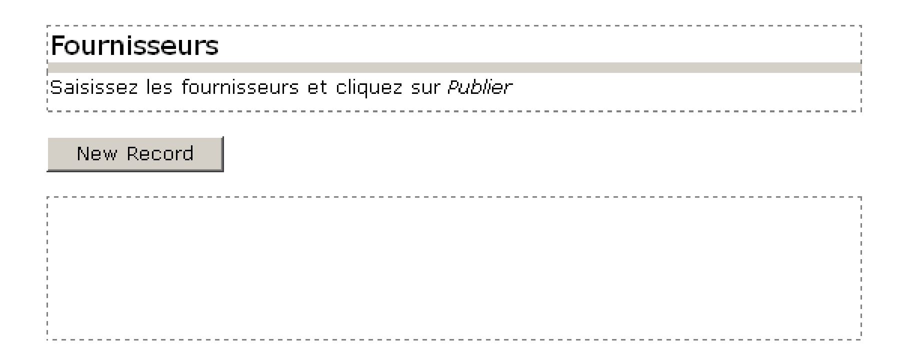
et vous devez avoir:



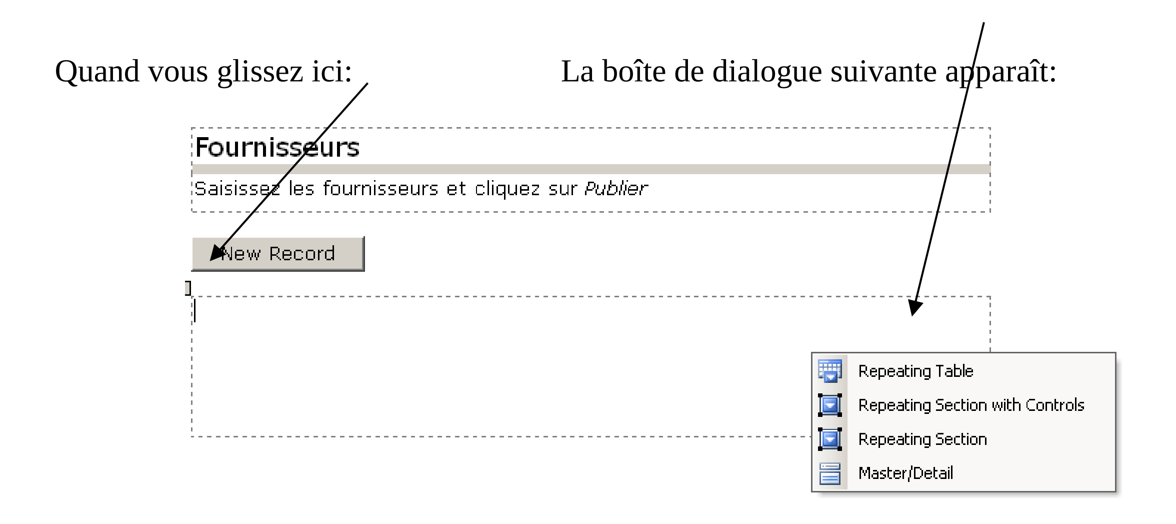
Cliquez sur *Next*, donnez un nom à votre connexion (*Fournisseurs* par exemple) et sur *Finish*. Vous devrez avoir le résultat suivant dans le volet Office:



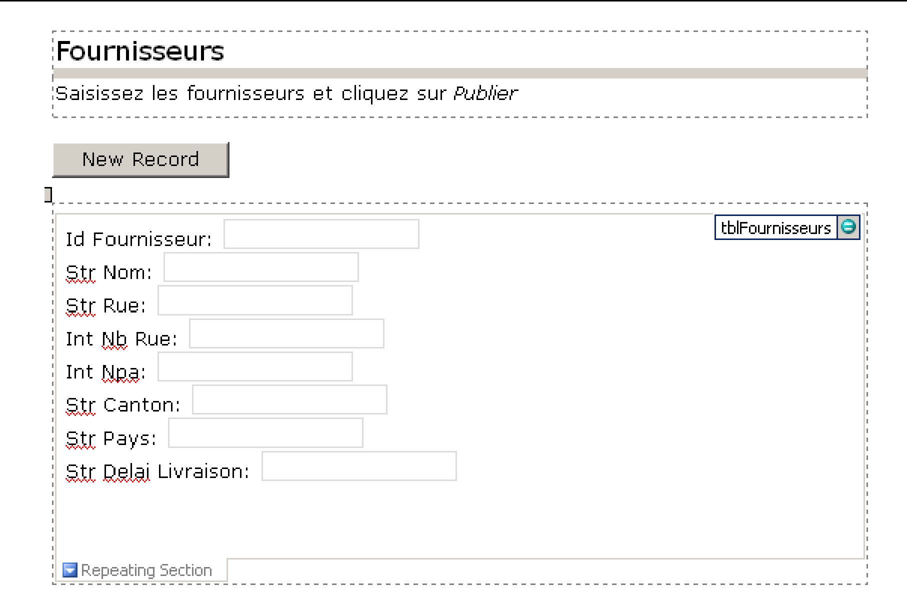
A gauche sur la feuille (InfoPath SP1), vous avez une zone *Drag data fields here* (vous pouvez effacer la partie concernant la *Query*). Glissez-y le nœud *d:tblFournisseurs:*



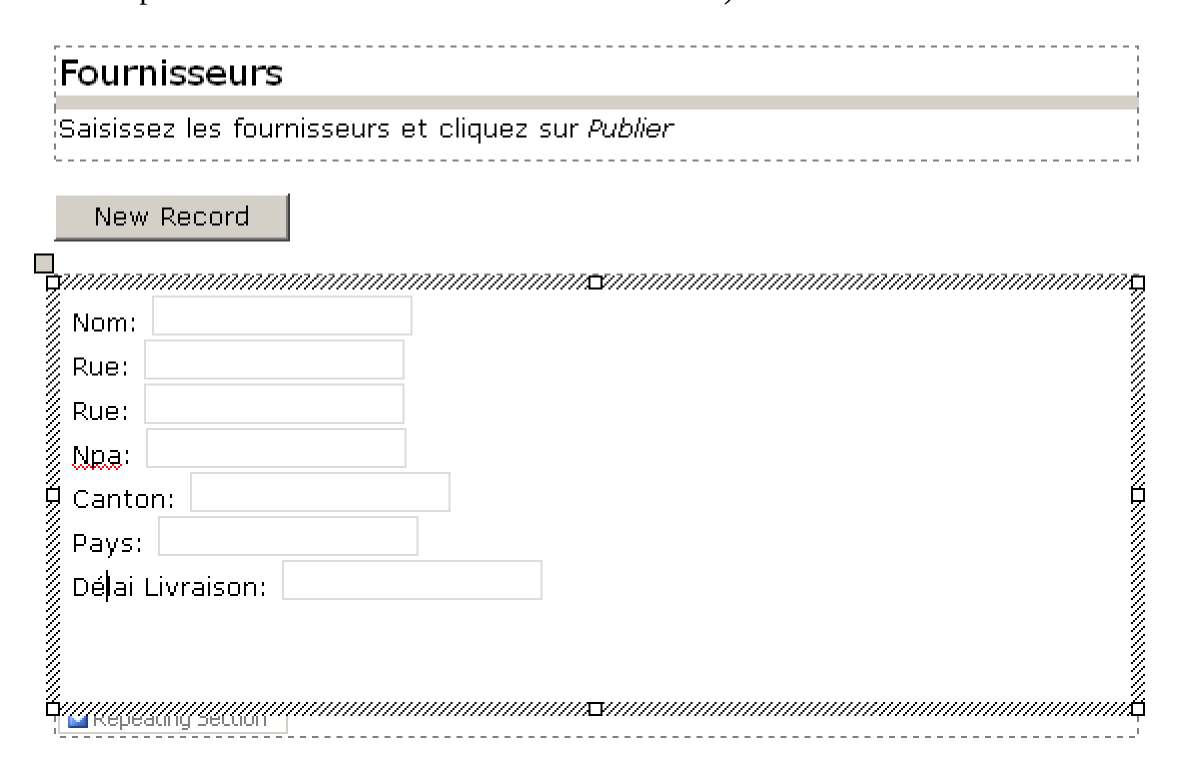
Quand vous glissez ici: La boîte de dialogue suivante apparaît:



Dans la boîte de dialogue, sélectionnez *Repeating Section with Controls.* Vous aurez:

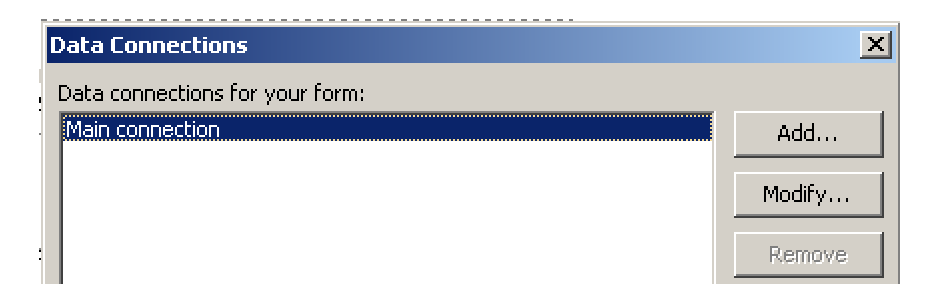


Vous pouvez changer le nom des étiquettes.et supprimer le champ *id Fournisseur* (puisque c'est un clé primaire *AutoNumber* dans notre base Access):

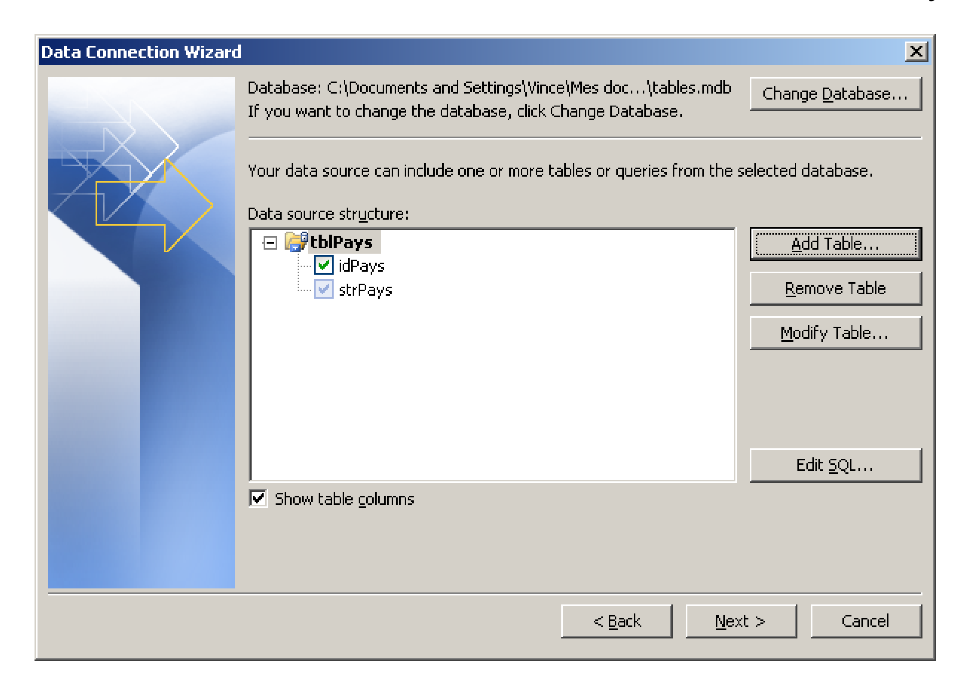


Maintenant nous aimerions que lorsque l'utilisateur cliquer sur *Canton* ou *Pays* que la liste des cantons ou pays disponibles dans la base Access (dans les tables *tblCantons* et *tblPays*) s'affiche à l'écran (pour que l'utilisateur n'ait pas à les saisir).

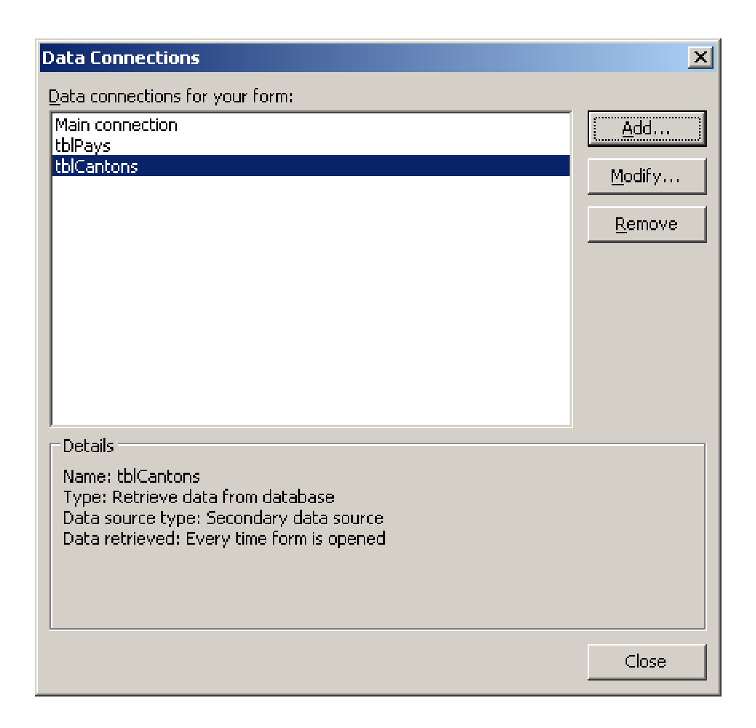
Pour ce faire, il faut aller dans le menu *Tools/Data Connection*:



*Add…*ensuite *Receive Data* ensuite *Database (Microsoft SQL Server or Microsoft Access only)* ensuite *Select Database* et allez chercher la base *tables.mdb* et la table *tblPays*:



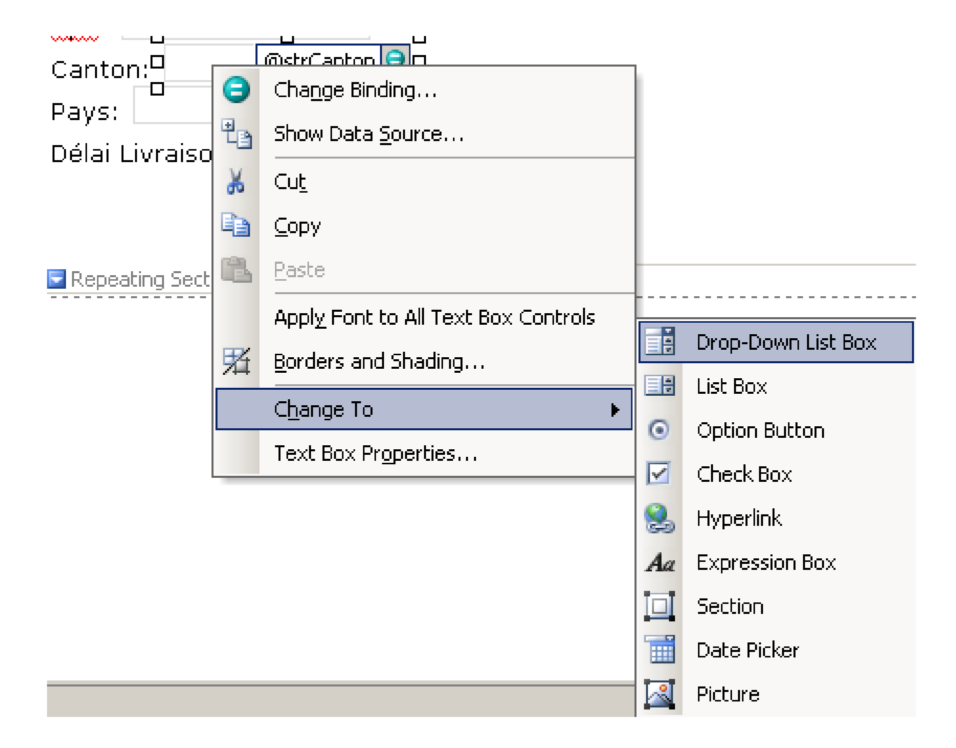
Cliquez sur *Next*, donnez un nom à la connexion et recommencez l'opération pour la table des cantons. Vous aurez alors dans la fenêtre des connexions (*Tools/Data connections*) trois connexions:



Cliquez sur *Close*.

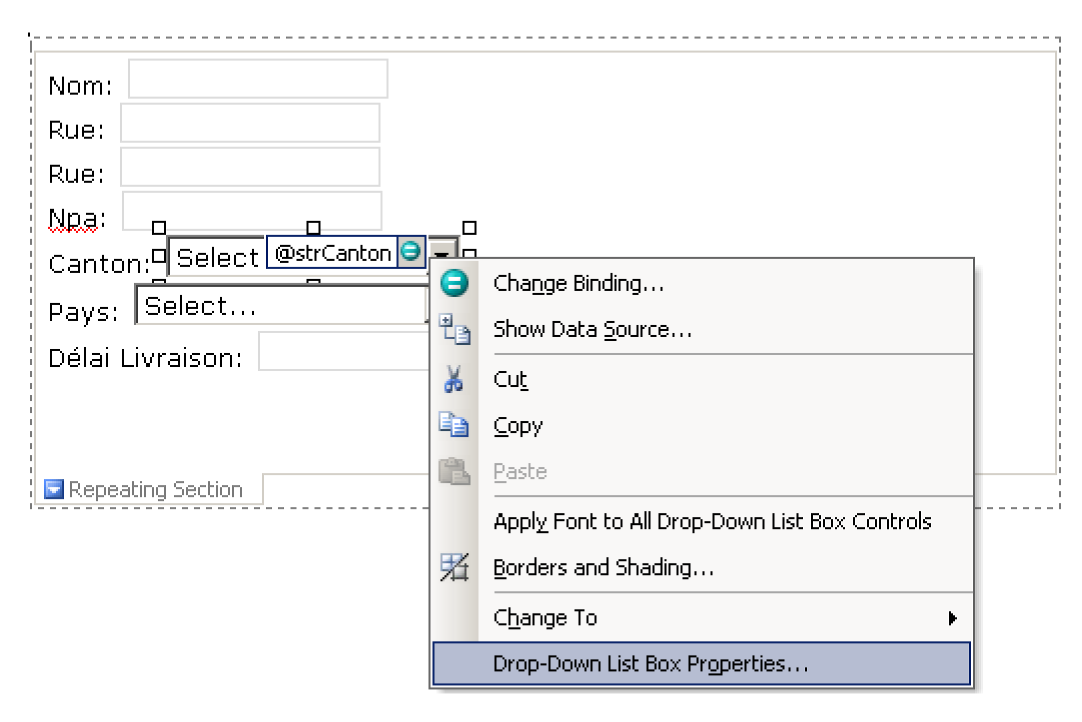
Pour relier les champs *Cantons* et *Pays* du formulaire InfoPath aux tables de la base Access et les transformer en listbox voici comment il faut procéder:

Il faut d'abord transformer les champs texte en Drop Down Listbox. Pour ce faire, bouton droit et:

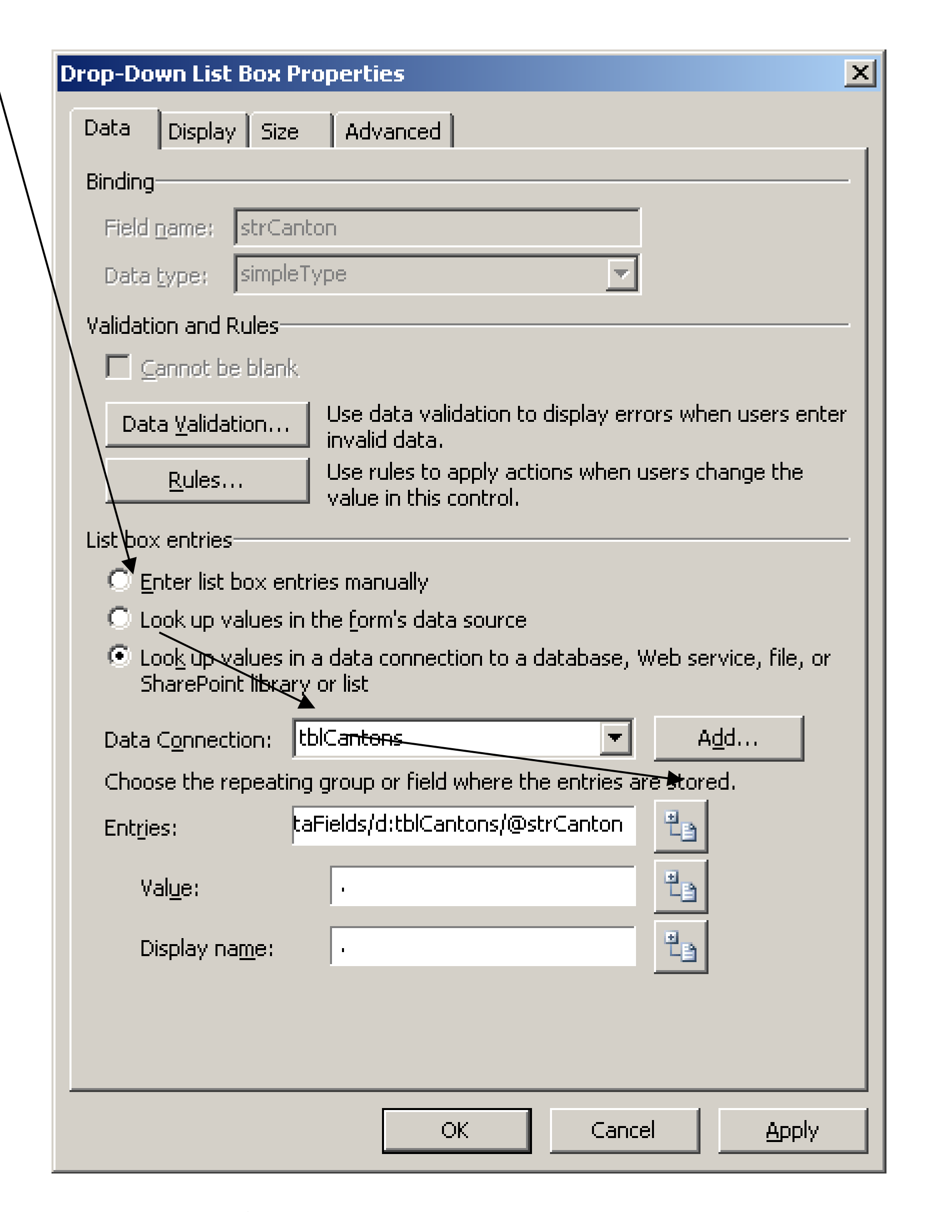


et idem pour les pays.

Maintenant il faut "peupler" les ListBox avec le contenu des tables de la base Access. Pour ce, clique droit et:

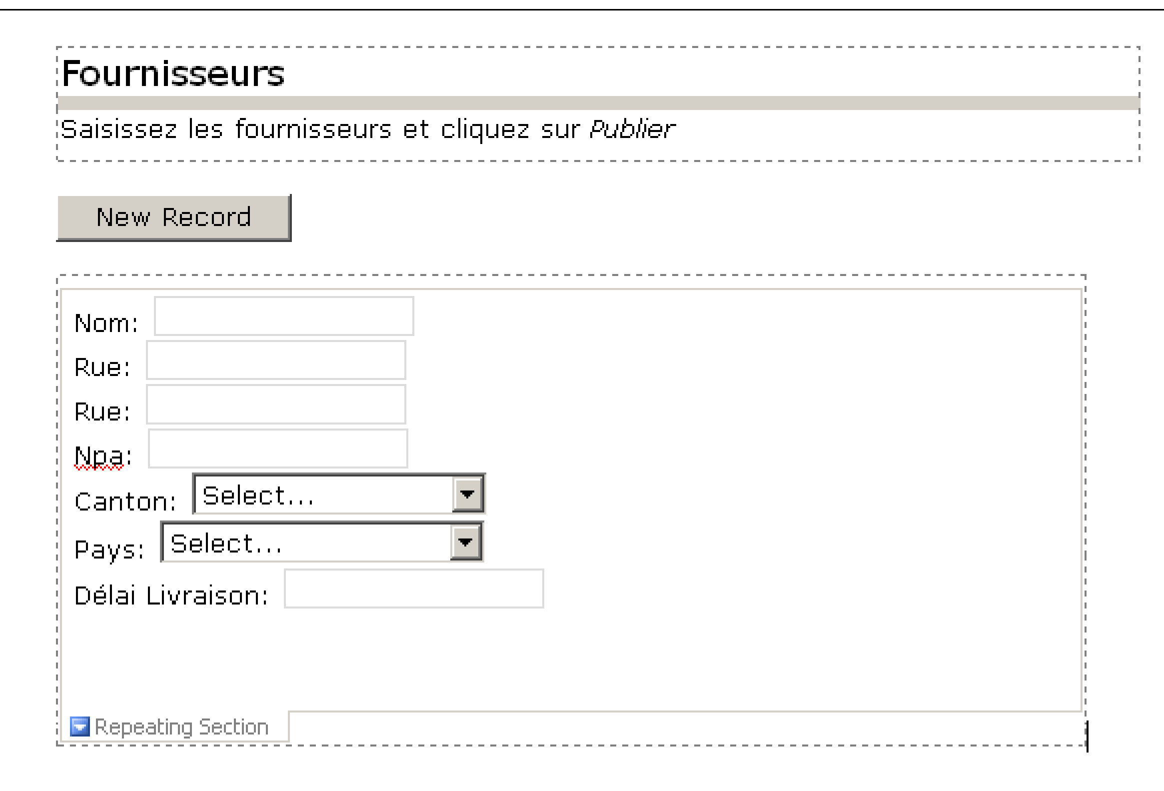


Ensuite:

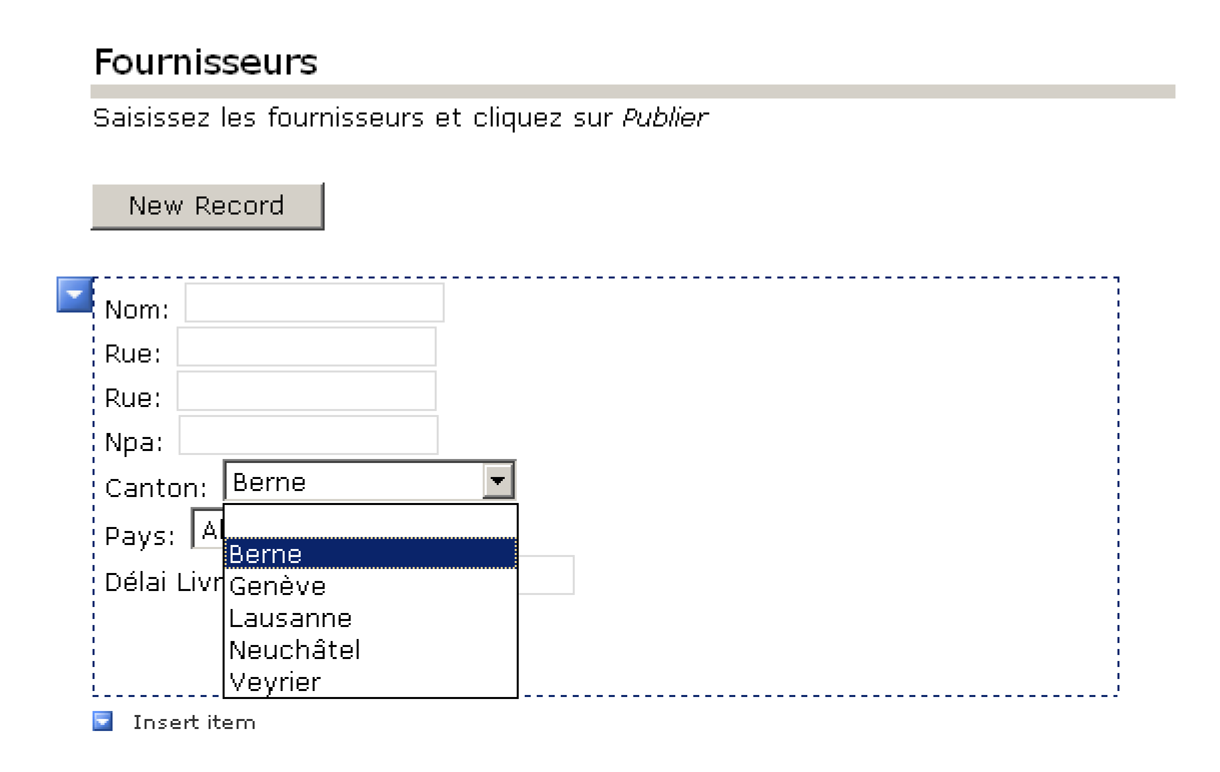


Vous cliquez sur *OK* et vous faites idem pour les pays.

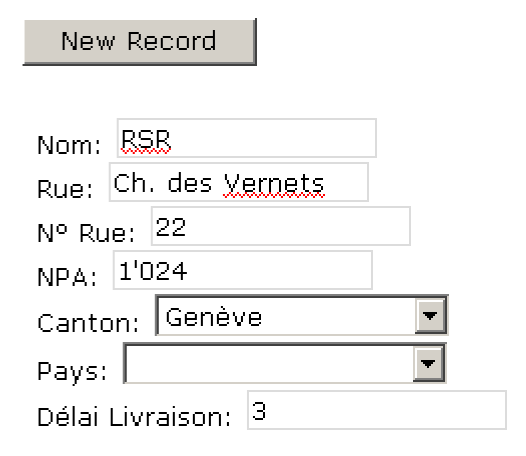
Vous enregistrez votre formulaire (*.xsn) et l'ouvrez par un double clique depuis votre raccourci sur le bureau ou votre explorateur Windows (si lorsque vous l'ouvrez il vous dit que les données se trouve sur un autre domaine vous validez par *Oui*):

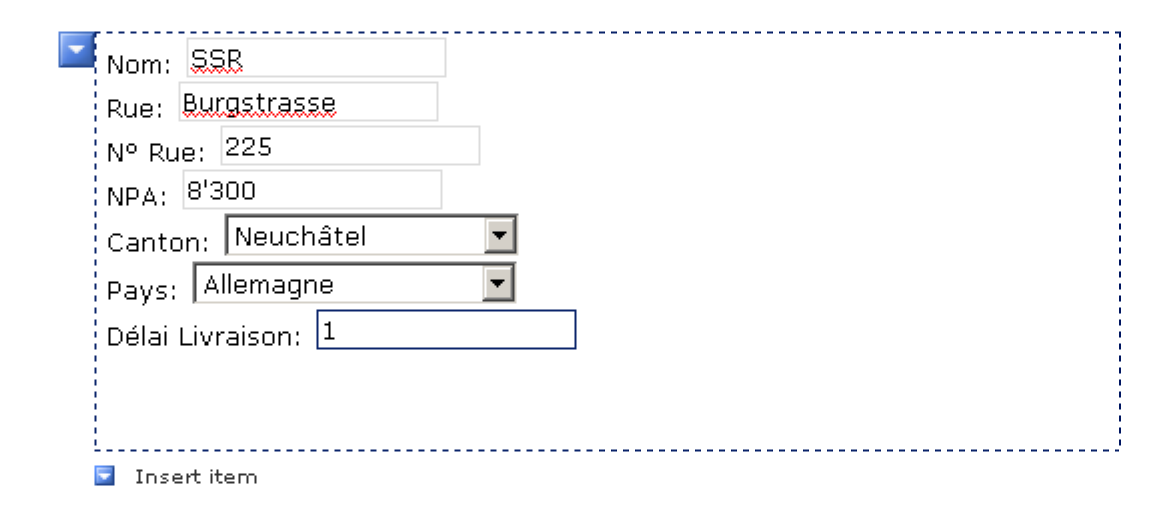


Cliquez sur une des ListBox et oh miracle:

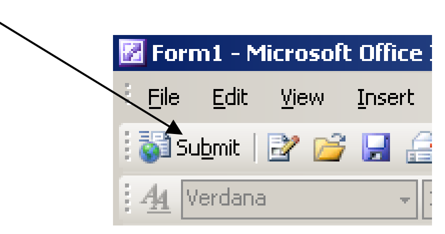


et maintenant pour la saisie de données ? Eh ben c'est très simple, on saisit le premier *Record* et le deuxième (cliquez sur *Insert Item*) etc…:

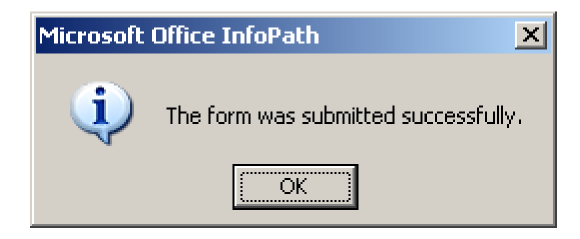




et quand c'est fini, on clique sur:

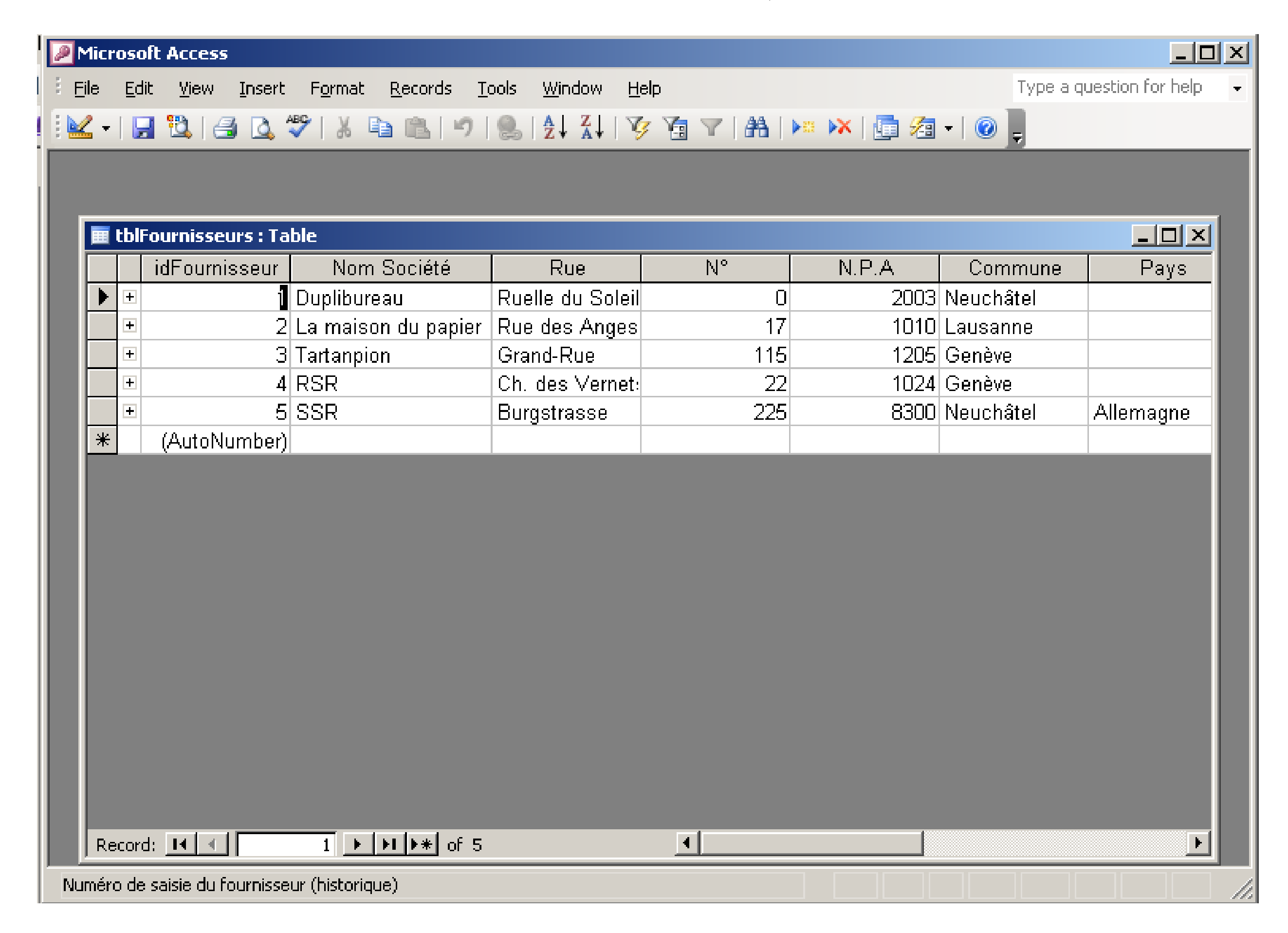


on a:



☺ on est forts hein…

Et si nous allons ouvrir notre table Access *tblFournisseurs*, nous avons bien:



Remarque: pour supprimer le bouton d'édition de MS InfoPath sur les postes clients allez dans *Tools/Options* et activez *Enable protection*.

Voilà pour les bases de MS InfoPath et XML.

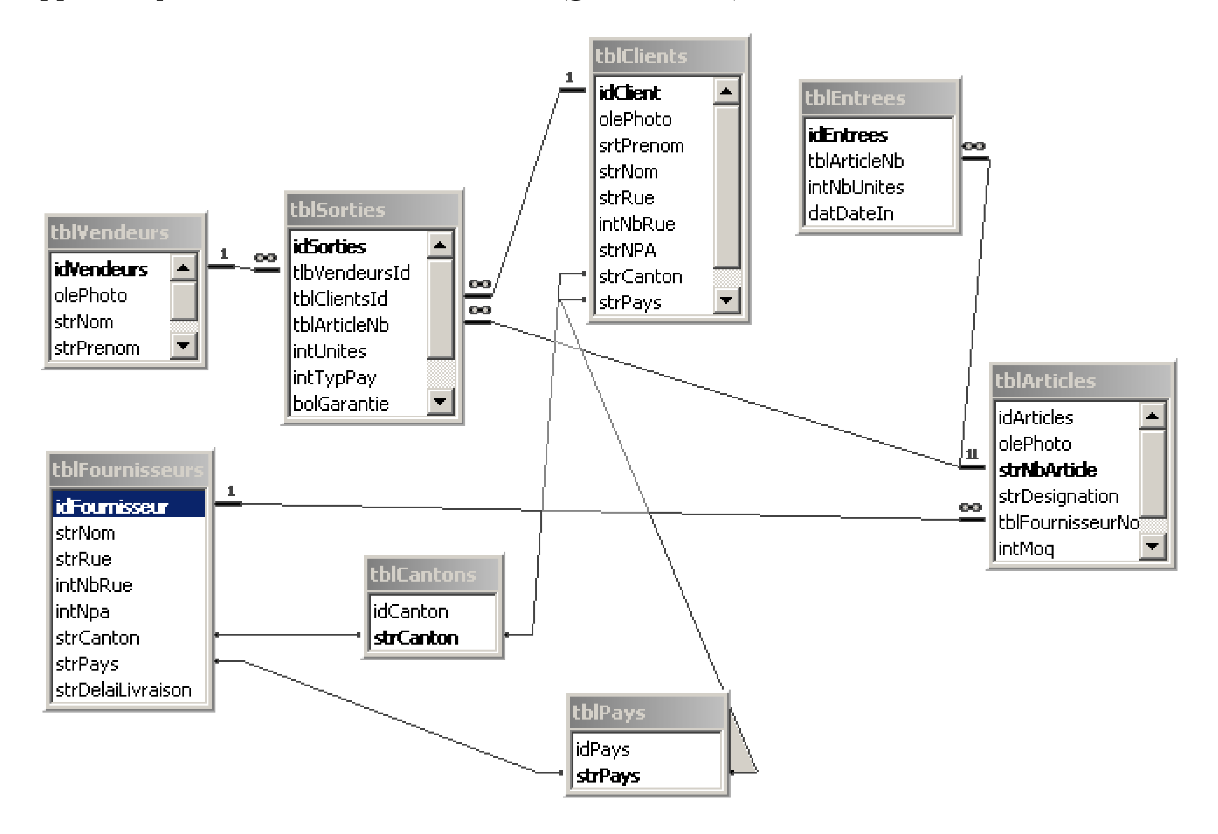
Bien sûr il faut avoir en tête que tout cela communique avec l'ensemble des produits de la gamme MS Office 2003.

27 MS SQL Server

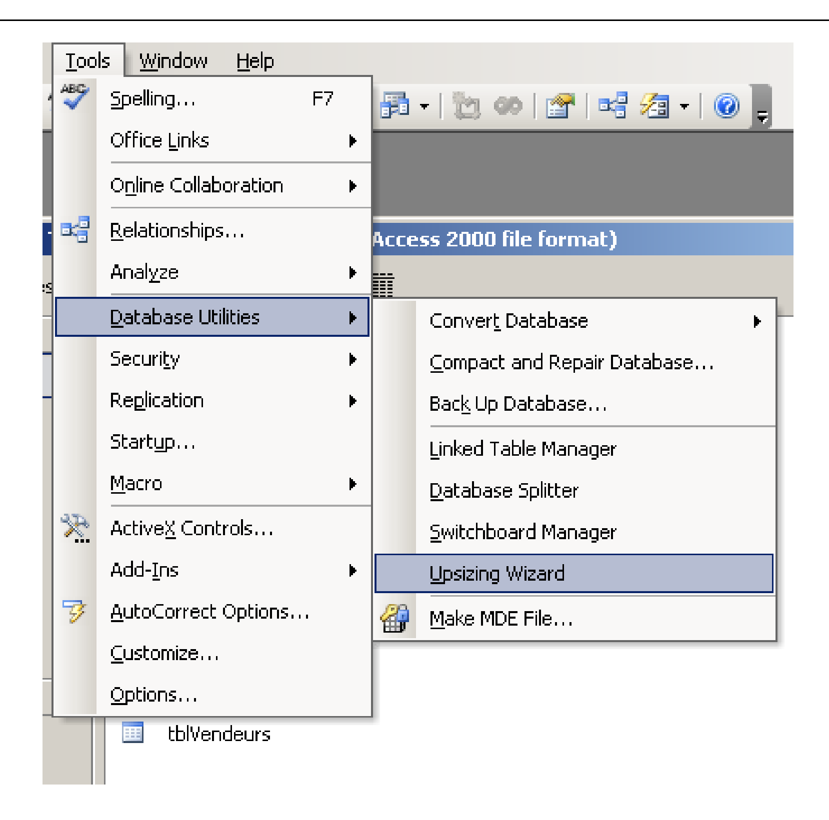
27.1 Reverse engineering vers SQL server

Nous allons voir maintenant comment envoyer notre base de données MS Access vers SQL Server. Pour l'exemple, nous disposons de MS Access 2003 en anglais, de notre base *Magasin.mdb* habituelle et de MS SQL Server 2000 SP3 (le serveur s'appelant "ISOZ"…).

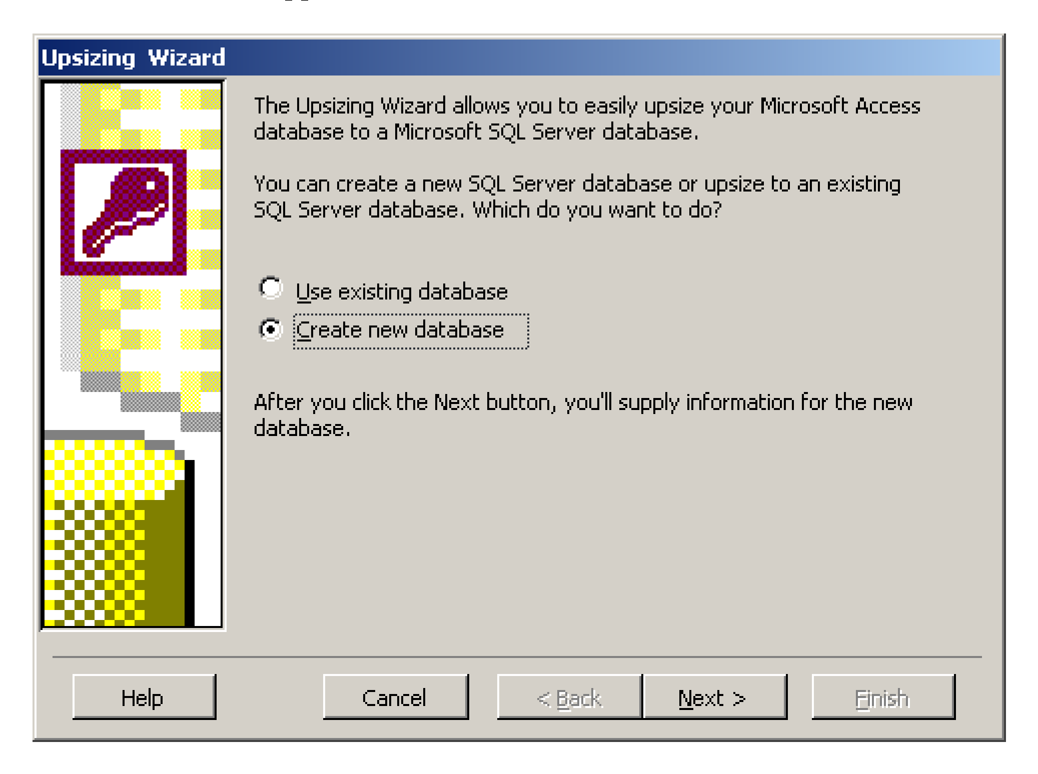
Rappelons que le schéma de notre base est (grosso modo) le suivant:



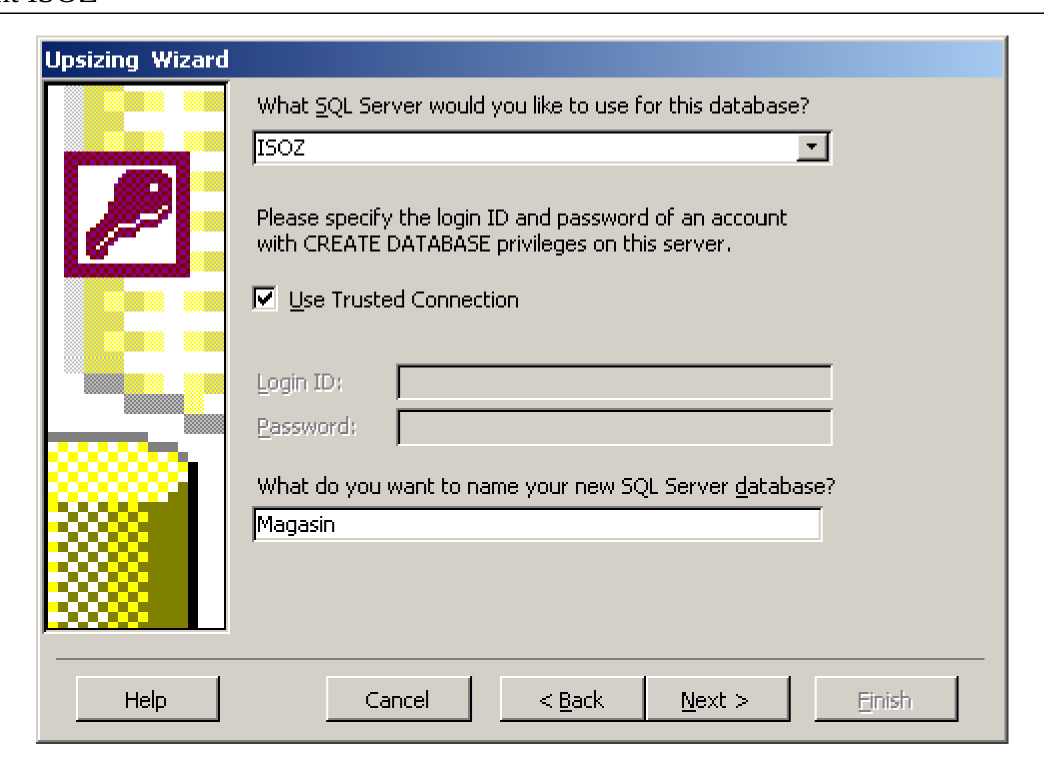
Pour la migrer vers MS SQL Server nous allons dans le menu suivant:



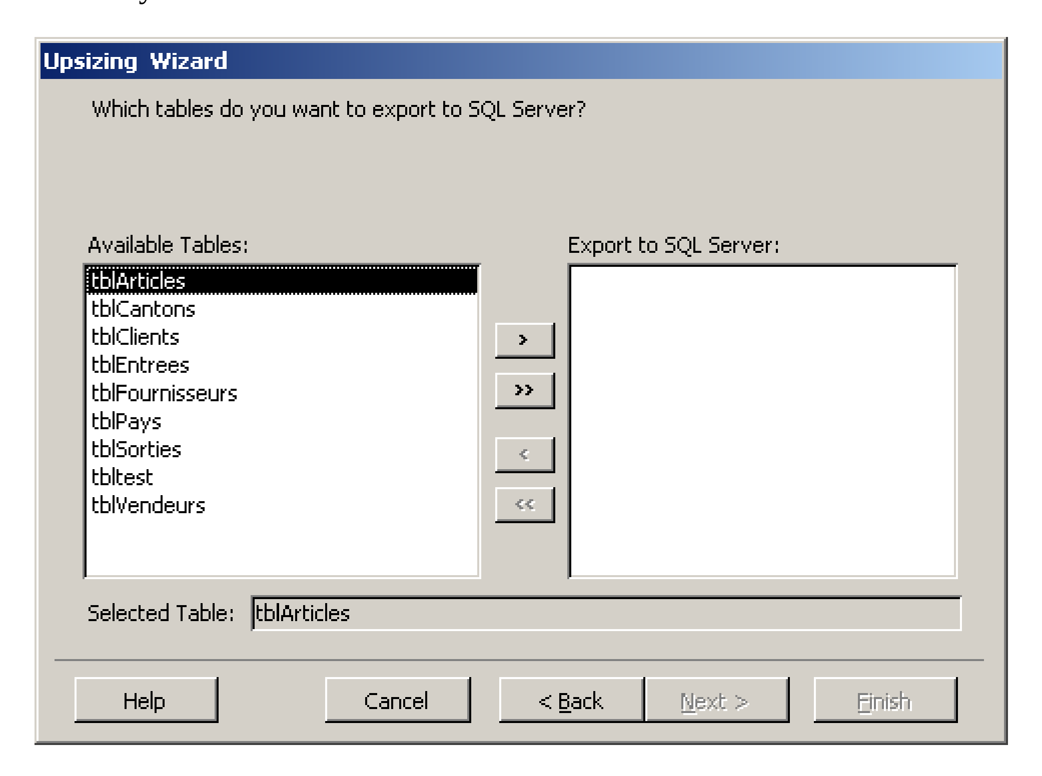
Dès lors la fenêtre suivante apparaît:



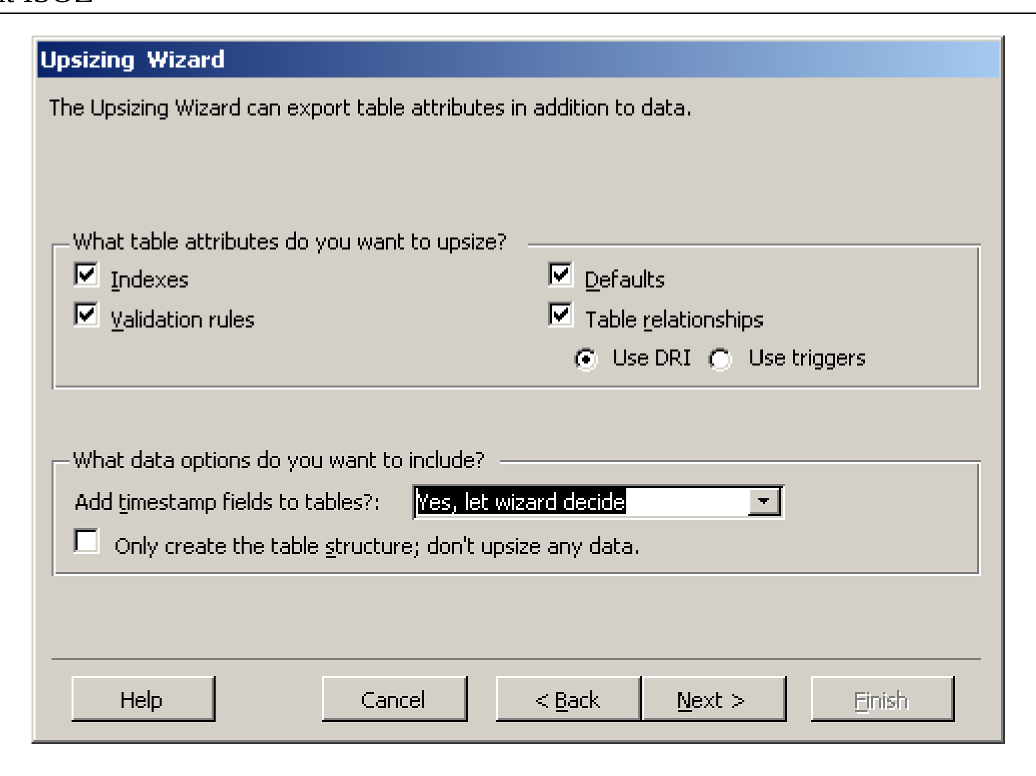
Nous allons donc créer une nouvelle base de données sur notre Server SQL et nous cliquons sur *Next*. La fenêtre suivante apparaît où nous saisissons le nom du Serveur et le nom de la future base (aucun paramètre de sécurité n'a été défini):



Nous cliquons alors à nouveau sur *Next*. MS Access nous demande alors quelles tables doivent être envoyées vers le serveur:



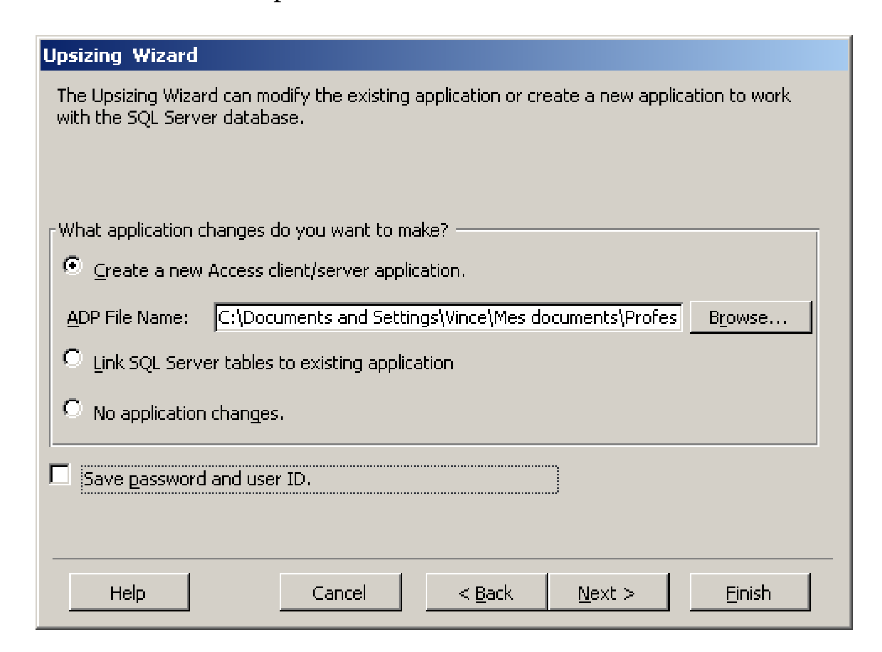
Nous les prenons bien évidemment toutes et cliquons à nouveau sur *Next*. La fenêtre suivante apparaît demandant si nous souhaitons prendre quelques propriétés particulières de la base et certaines manières de gérer les relations entre les données:



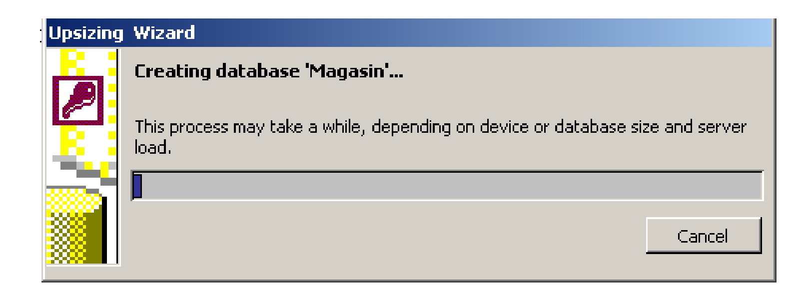
Sans rien changer cliquez sur *Next*. La prochaine fenêtre est très intéressant elle nous permet:

1. D'automatique lier le fichier *.mdb en cours afin de travailler sans presque rien changer aux codes et requêtes existantes

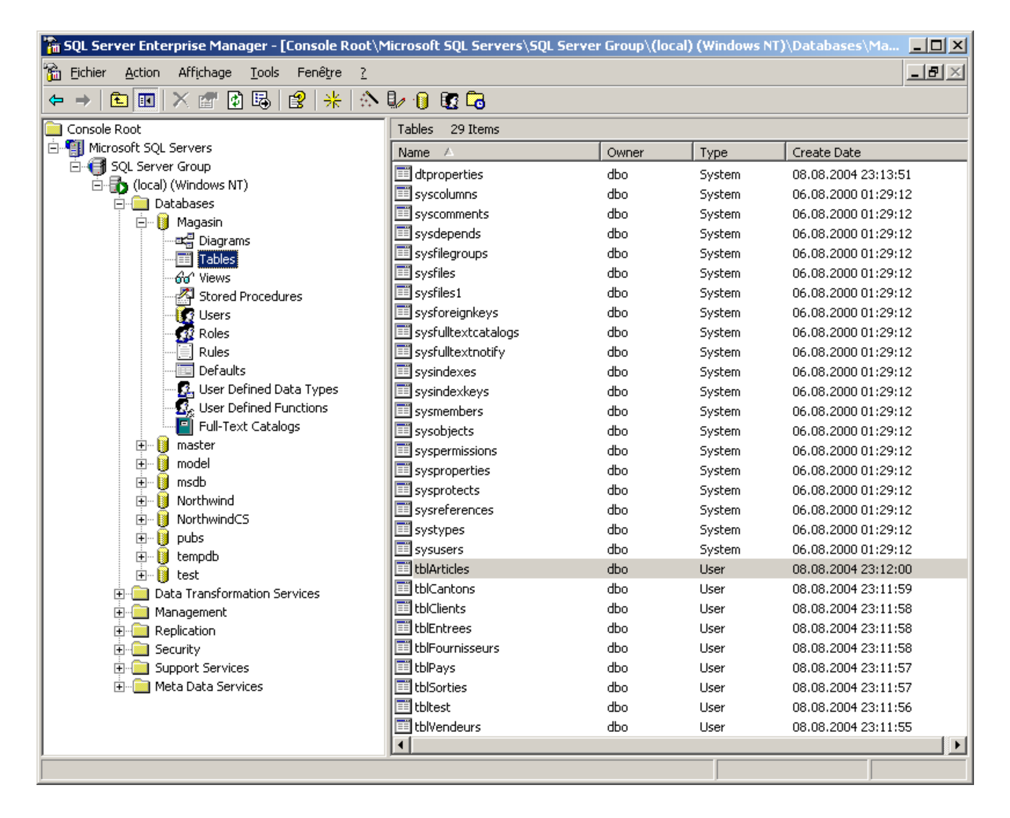
2. La seconde propose de créer un fichier *.adb c'est-à-dire les fichiers d'accès aux données dont nous avions parlé tout au début de ce document.



Sans rien changer, cliquez sur *Finish*. L'ordinateur va alors travailler pendant un petit moment:



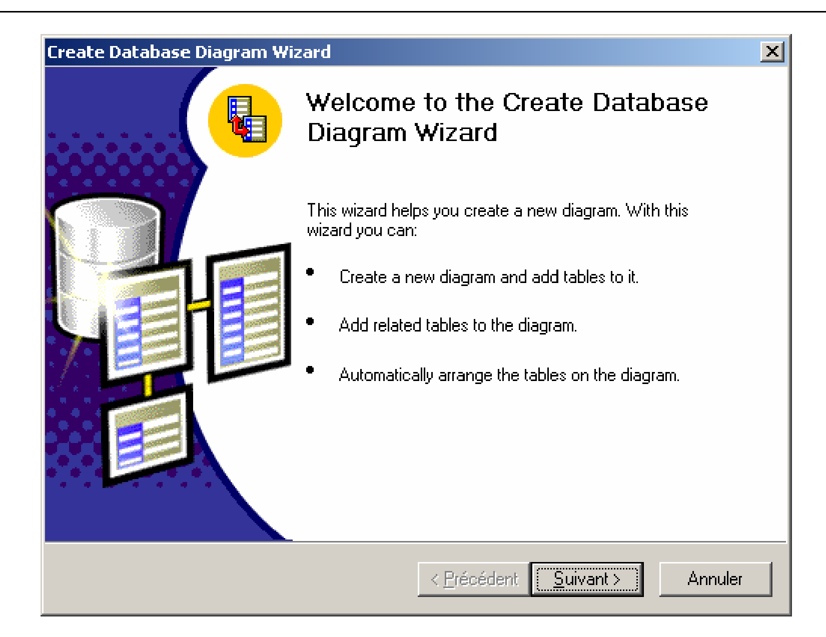
Ensuite il va vous afficher un rapport, fermer votre fichier *.mdb, ouvrir automatiquement le fichier *.adp et voilà ce que vous aurez dans SQL Server Entreprise Manager:



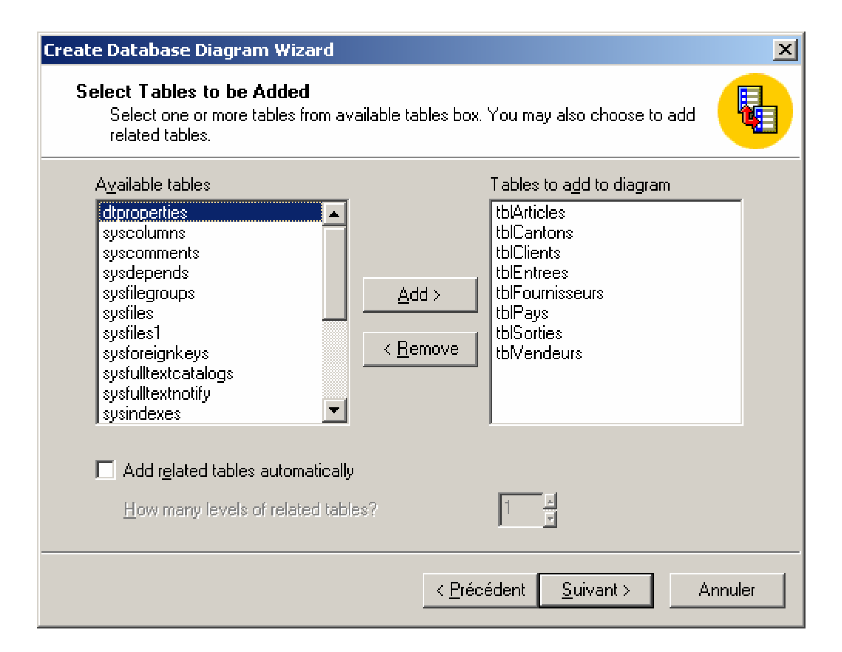
Nous y retrouvons donc bien notre base *Magasin* avec toutes ses tables plus de nombreuses tables relatives aux propriétés de sécurité de MS Access.

Voyons quelque chose de pertinant maintenant:

Création d'une nouvelle vue à partir du dossier *Diagrams* (bouton droit de la souris dessus et *New Diagram*):



Cliquez sur *Suivant:*



La fenêtre suivante apparaît:

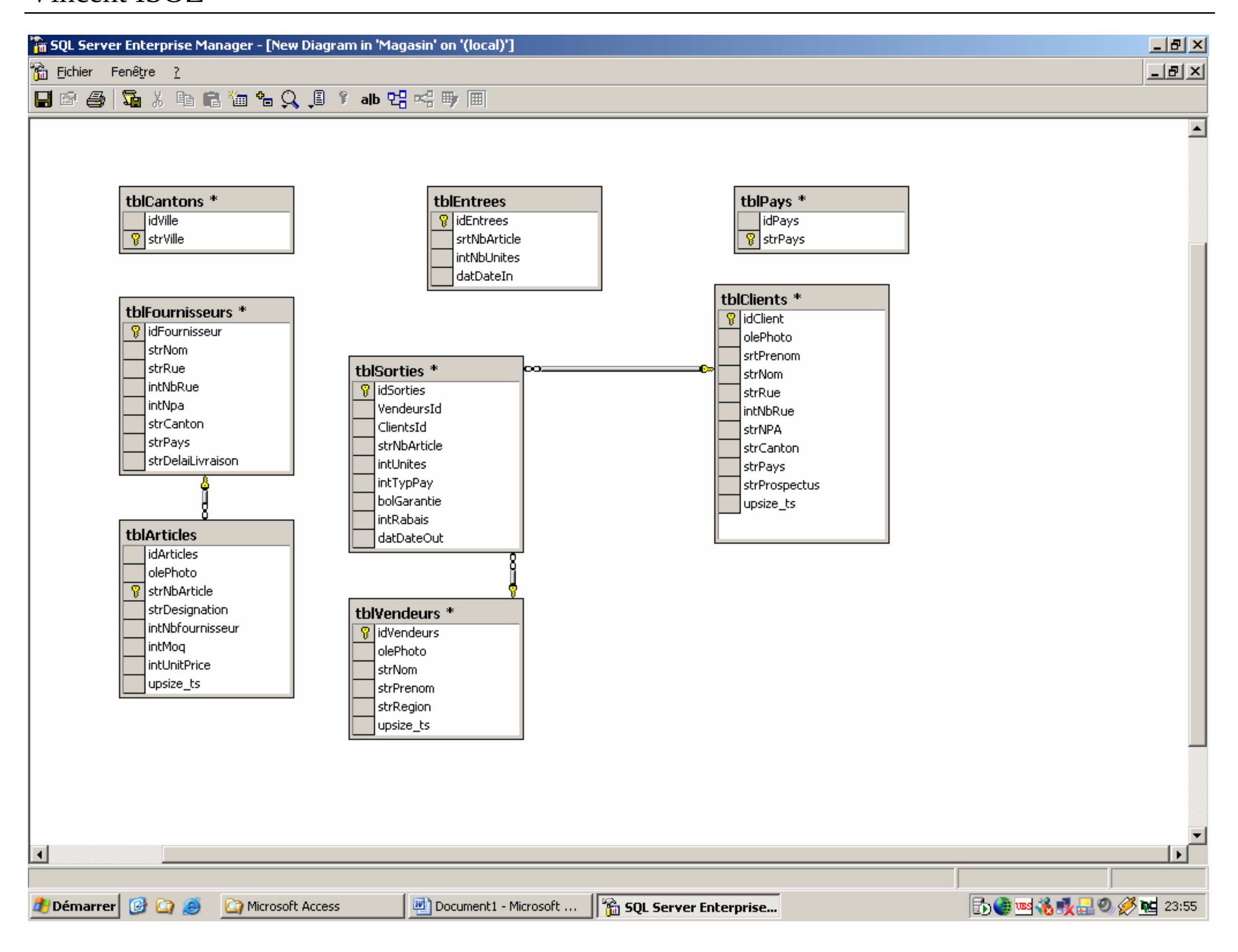
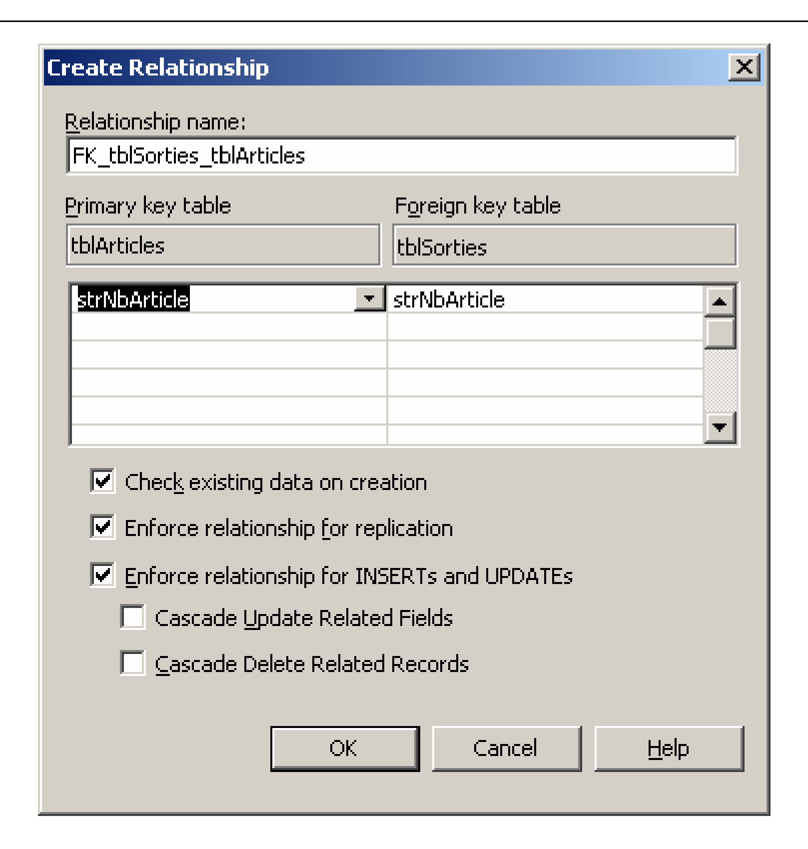
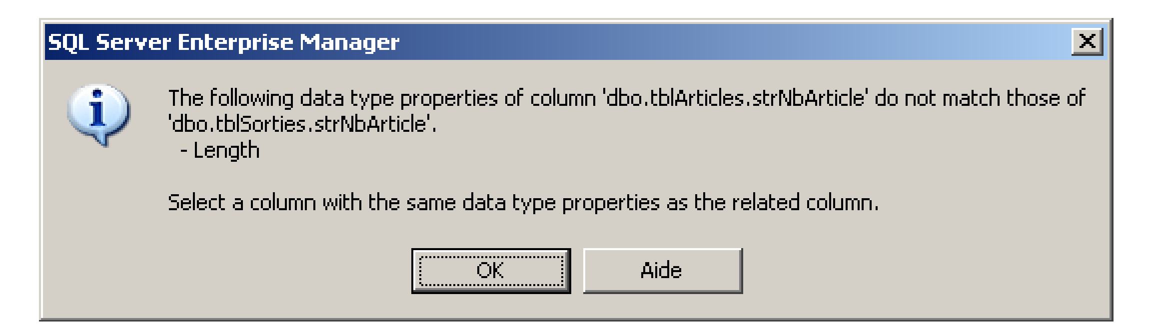


Figure 11 Schéma SQL Server

Nous voyons quelque chose d'intéressant: toutes les relations qui n'utilisent pas l'ID auto-number ne sont pas reconnues dans SQL Server d'où la tout première remarque que nous avions fait au début. Même si nous essayons de forcer les relations:



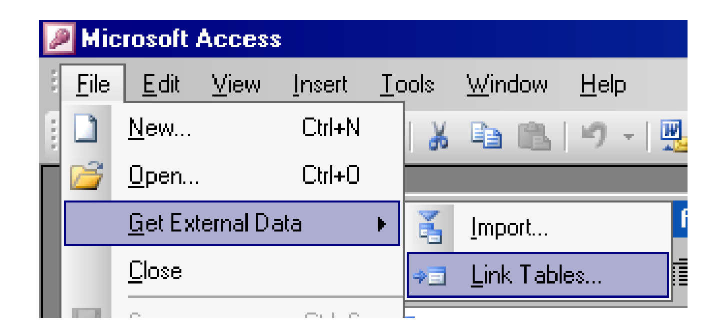
Problème évident:



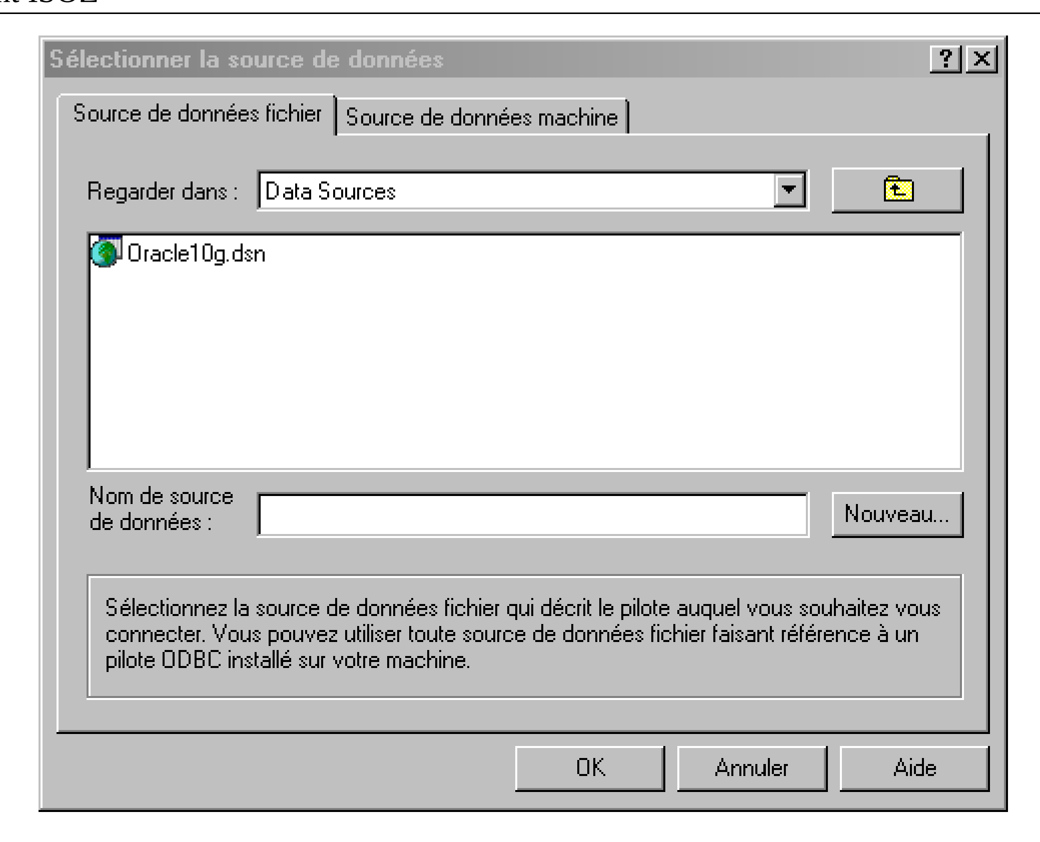
27.2 Connexion à SQL server

Nous allons voir ici comment nous connecter à une base de données SQL Server pour rapatrier des informations dans une base Access du type *.mdb existante.

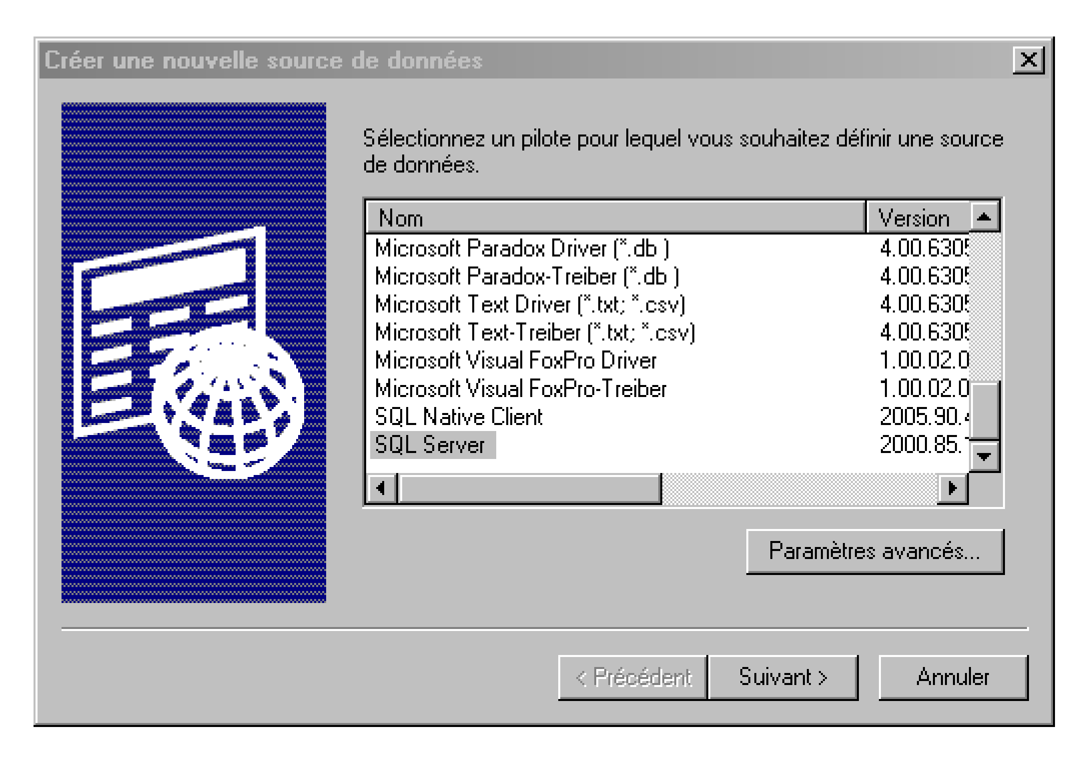
Pour cela il faut aller dans *Fichier/Données Externes/Lier les tables*:



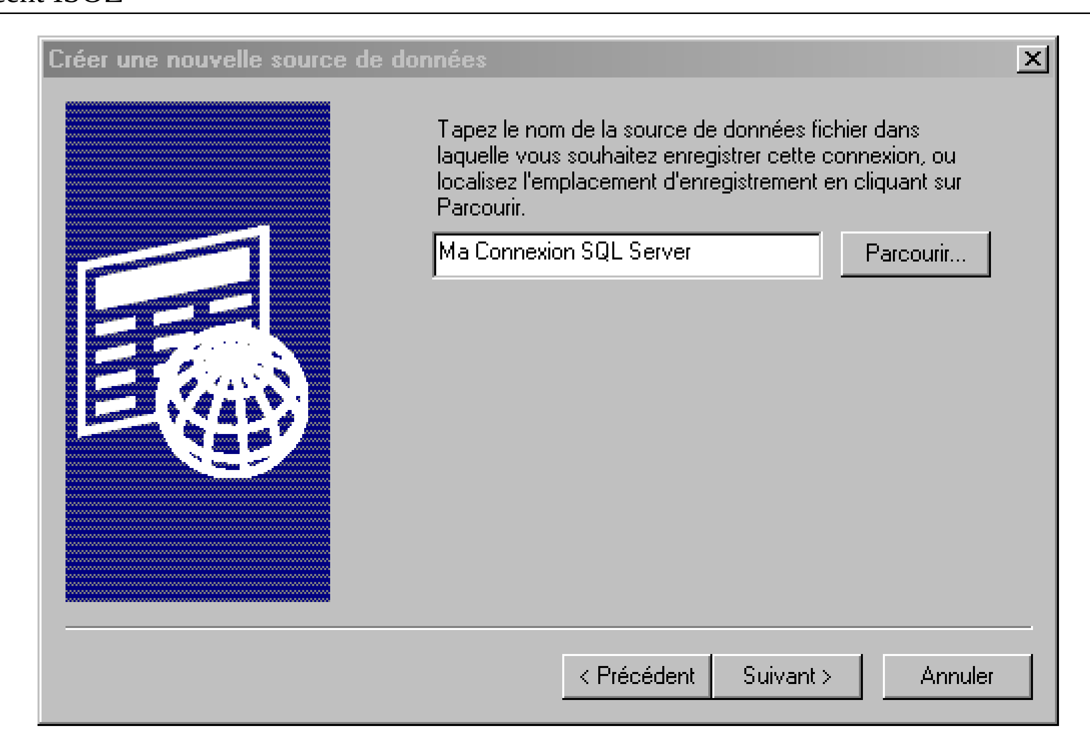
et choisir l'option ODBC dans la liste déroulante *Type de fichier*. apparaît alors automatiquement la boîte de dialogue suivante:



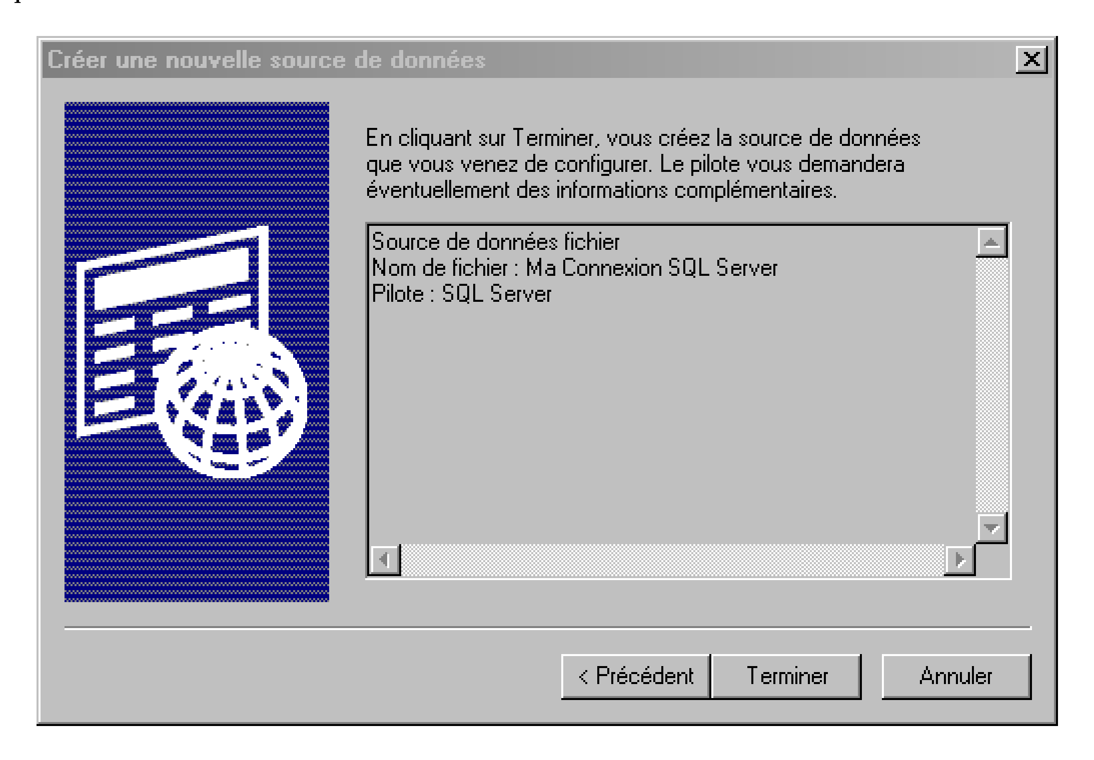
Cliquez sur *Nouveau*:



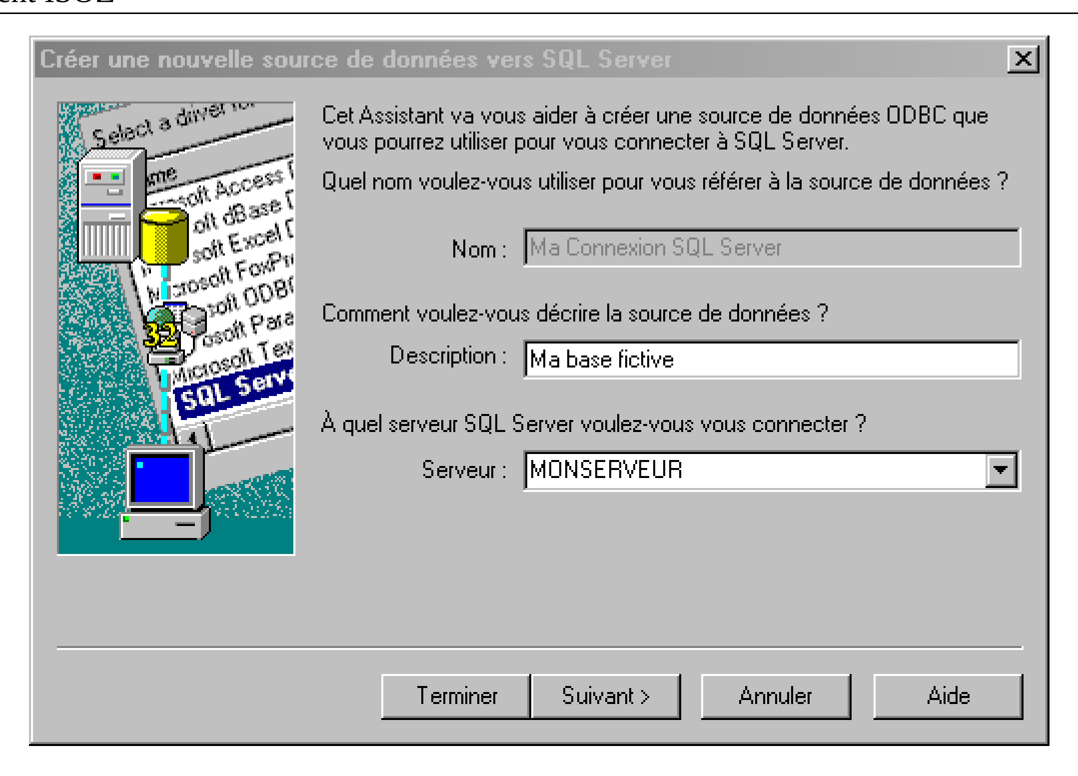
et choisissez le pilote *SQL Server* et cliquez sur *Suivant*. Ensuite, donnez un nom à la connexion:



et cliquez sur *Suivant*:



et sur *Terminer*. Vous arrivez alors à la boite de dialogue suivante:



bien évidemment il faut connaître les informations à saisir… Ensuite faites *Suivant* et laissez (car c'est le cas majoritaire) les options par défaut:



et la suite nécessite une connexion serveur… sinon cela ne passera pas…

28 AS/400

Dans le genre base de données préhistoriques (…) il existe une technologie appelée AS/400 encore utilisée par certaines entreprises (qui n'ont aucun intérêt à changer et elles ont bien raison!).

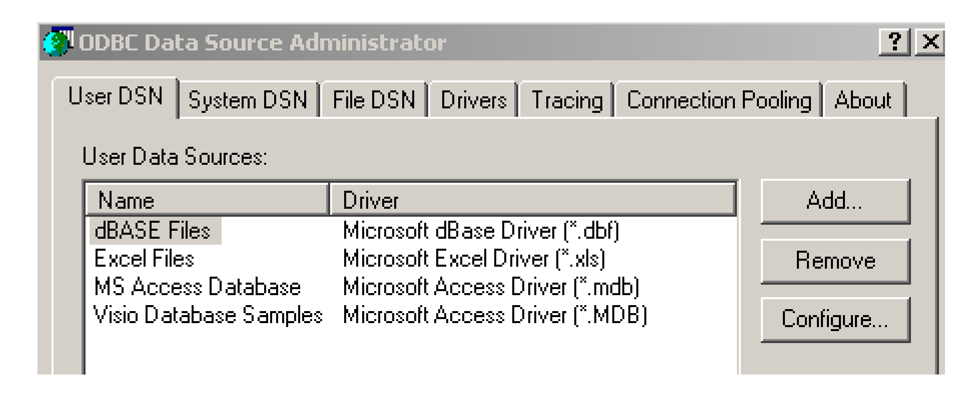
Pour ceux qui n'ont jamais vu à quoi cela ressemble, en voici une image:



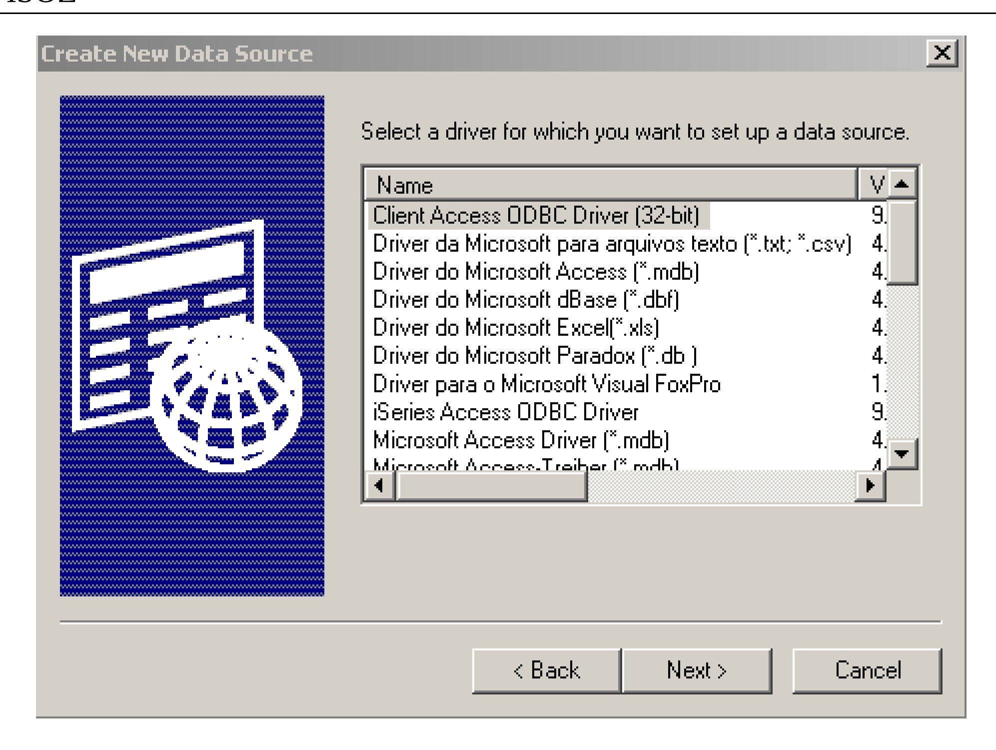
Ce que nous allons voir maintenant c'est comment se connecter à une telle base de données. Dans le panneau de configuration cliquez sur:



Ensuite, sélectionnez *dBase* comme ci-dessous:



Cliquez sur *Add…*et sélectionnez le driver mis en évidence ci-desous:



Attention! Si vous n'avez pas ce driver c'est que soit:

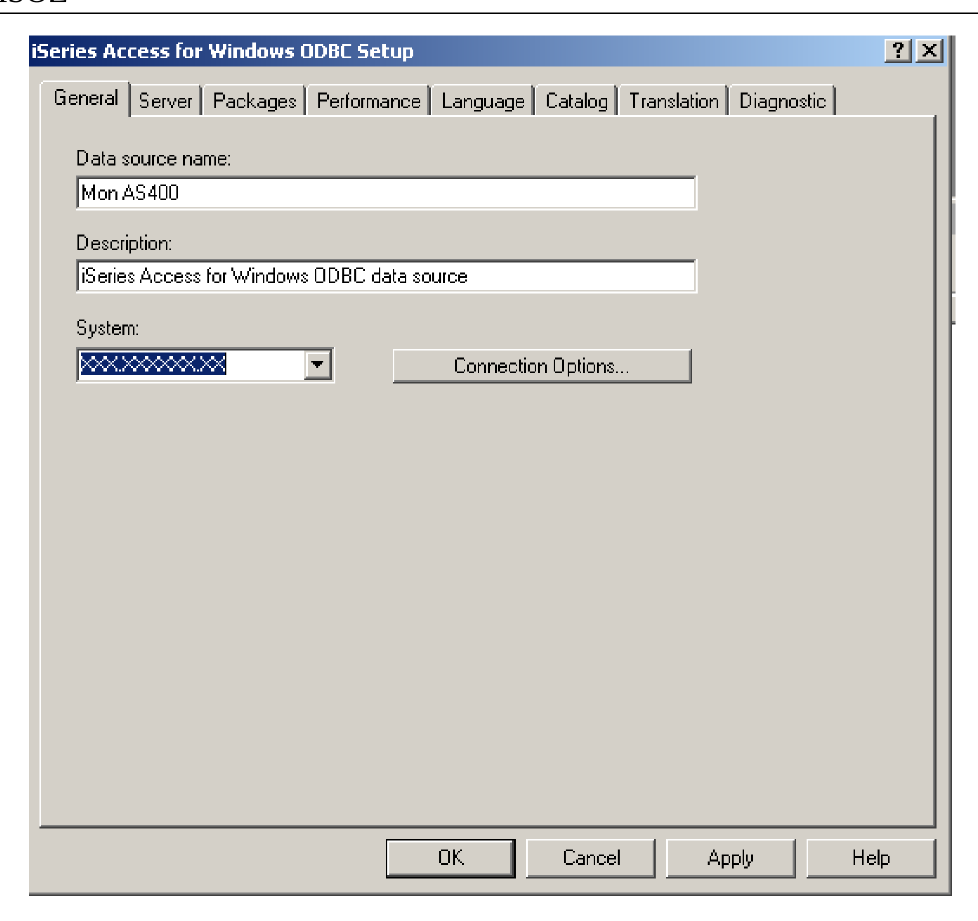
1. Vous n'avez pas installé Microsoft Access avec toutes les options

2. Que votre ordinateur n'est pas à jour!

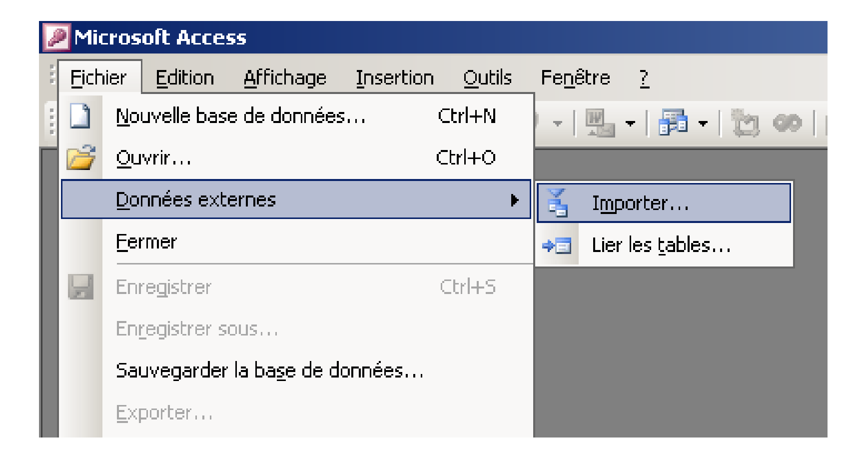
3. Votre ordinateur ne reconnaît pas que dans le domaine il y a un AS400

Dans la fenêtre qui suivra (désolé mais je n'ai pas d'AS400…) il faudra dans l'onglet *General* saisir un nom pour la connexion ODBC (typiquement on choisira *AS400*) et on spécifiera l'adresse IP de l'AS400 dans le champ se situant sous la description de la connexion.

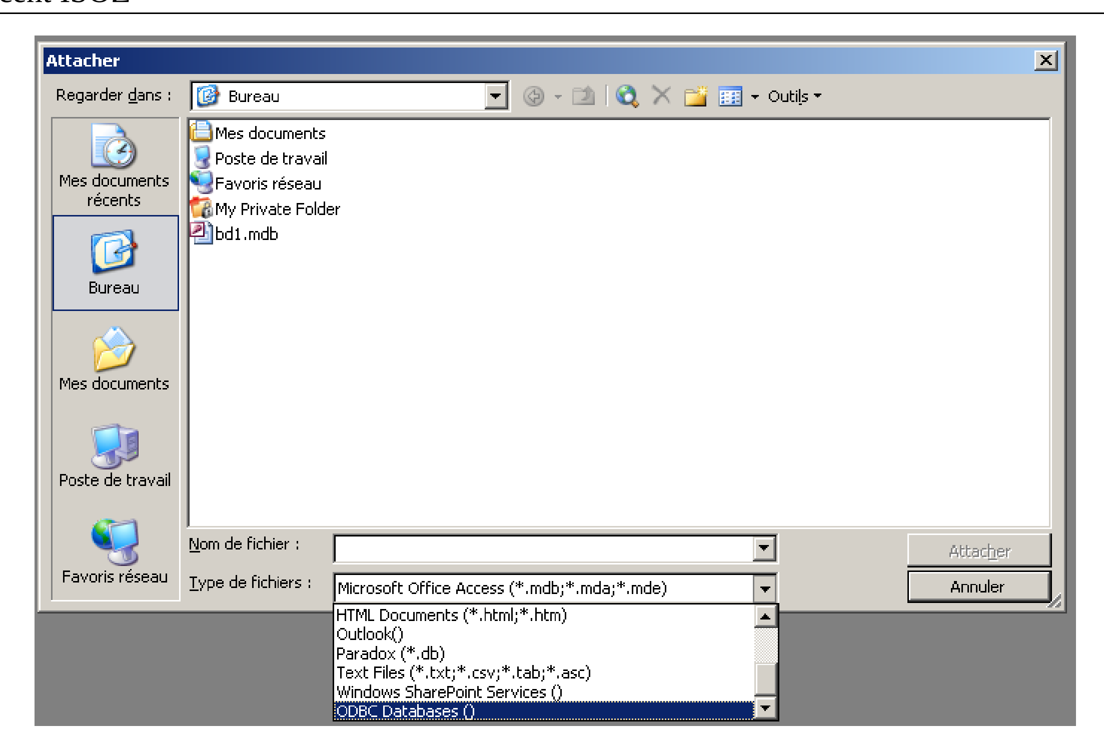
Attention! Enregistrez toujours une requête avant de l'exécuter.



Une fois ceci fait, en théorie il suffit de valider le tout et de retourner dans MS Access et aller dans *Fichier/Données externes* et choisir *Lier les tables…*:



et de choisir:



et dans l'assistant qui suite de sélectionner la connexion ODBC créée précédemment.

Remarque: Si vous enregistrez des requêtes AS400 au format ***.dtf** (générées avec iSeries) celles-ci ne peuvent malheureusement pas être connectées directement à MS Access… Il faut pour cela utiliser le fichier que génère le *.dtf, fichier choisi lors de la création du *.dtf même dans les options de l'AS400 (csv, txt, xls). S'il s'agit d'un fichier MS Excel, CSV ou TXt, reportez vous au chapitre correspondant Liaison MS Excel/CSV.

Pour mettre à jour automatiquement les requêtes *.dtf qui généreront les fichiers *.csv, *.txt ou *.xls vous pouvez passer par le code VBA suivant:

```
Option Compare Database

'La ligne ci-dessous peut s'avérer indispensable si la requête dtf importe des millions de
données et que du code s'exécute par la suite
Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

Sub ShellLogin( )

  OpenDtf=Shell("Nom du fichier.dtf", 1))

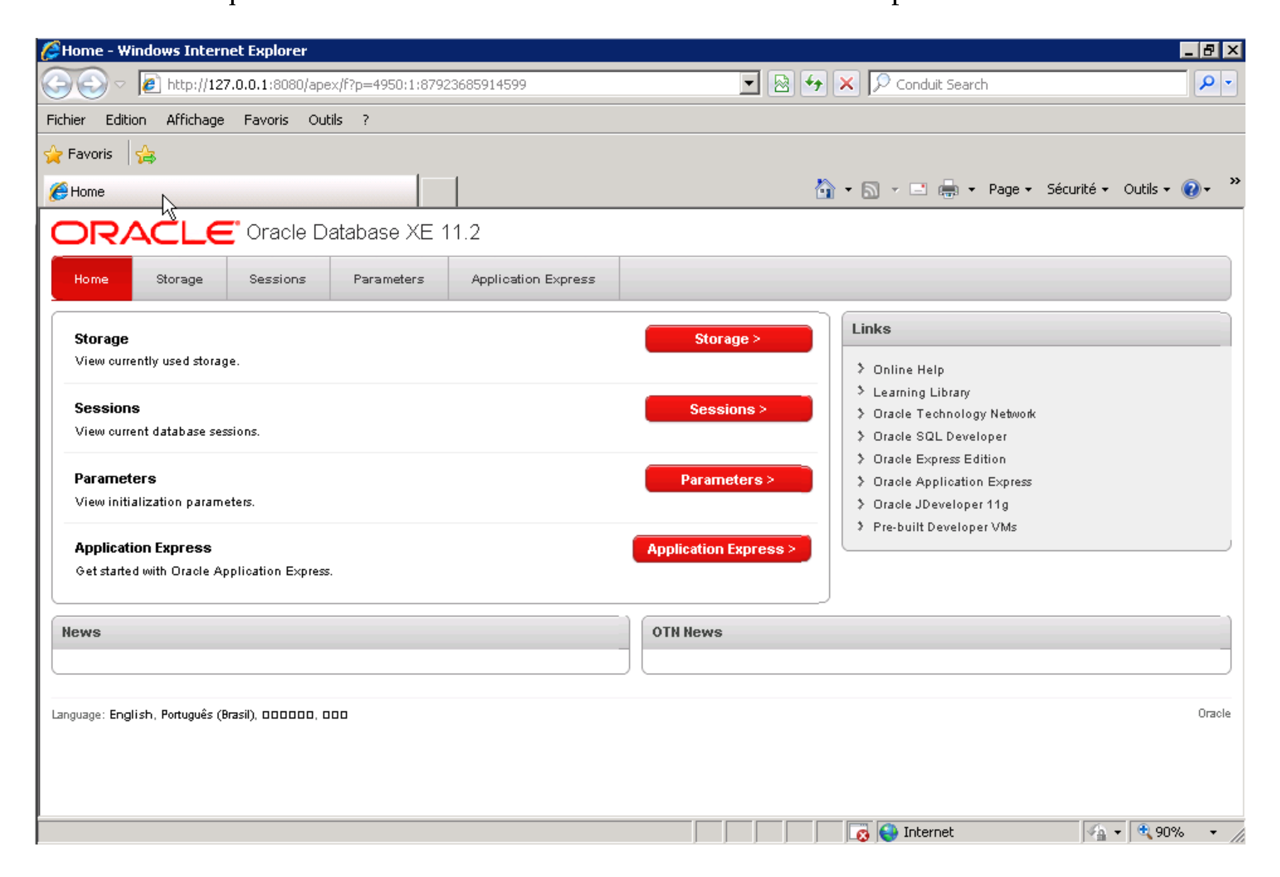
  Sleep 1000

End Sub
```

29 Oracle Express 11.2g

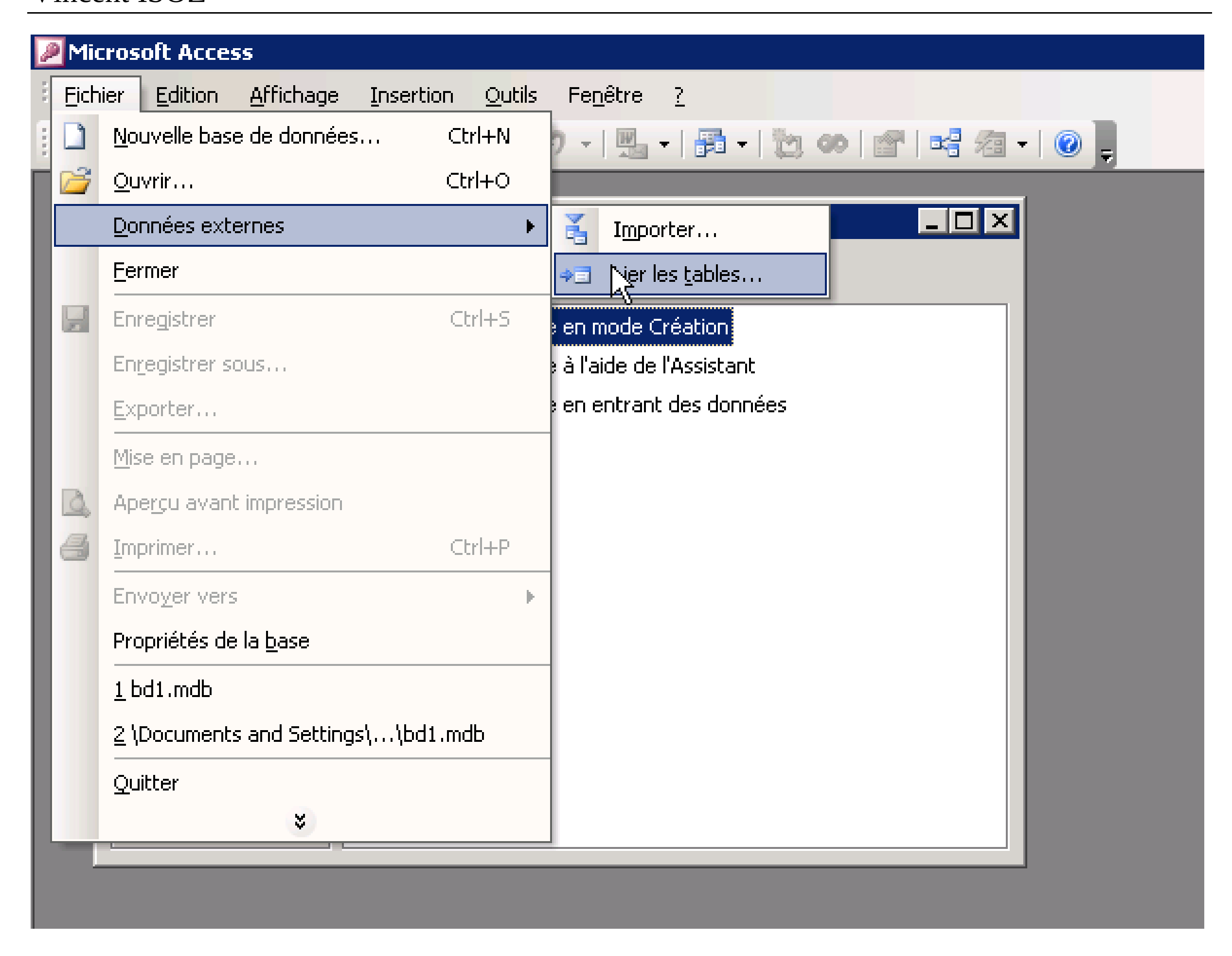
Nous allons voir ici comment nous connecter à une base Oracle Express 10g. Evidemment le système est toujours le même…

D'abord nous supposerons que le produit Oracle Database Express 11.2g est installé et fonctionnel et que vous connaissez votre nom utilisateur et mot de passe:

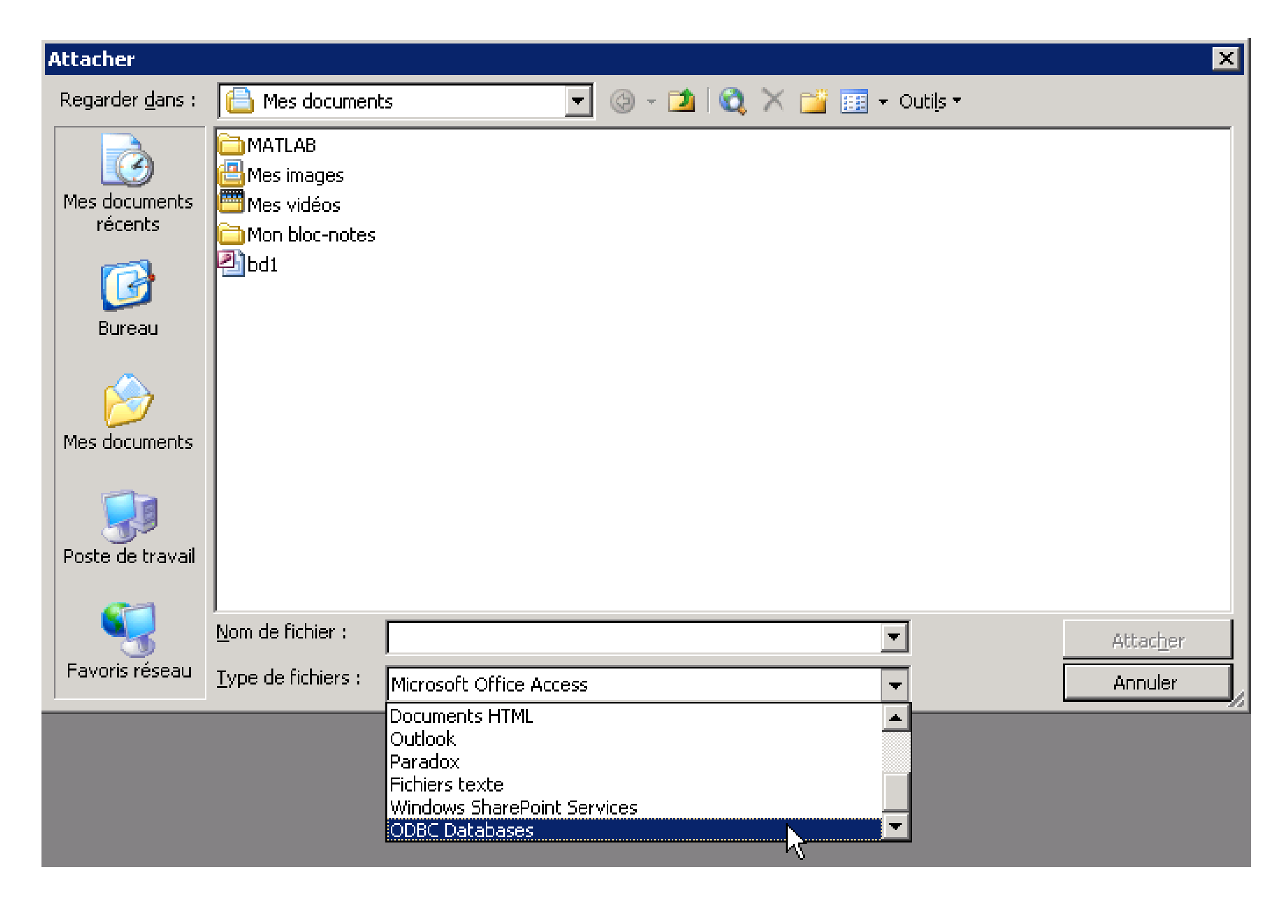


Dans notre installation, le nom d'utilisateur sera *system* et le mot de passe sera *root* (choisi au début de l'installation).

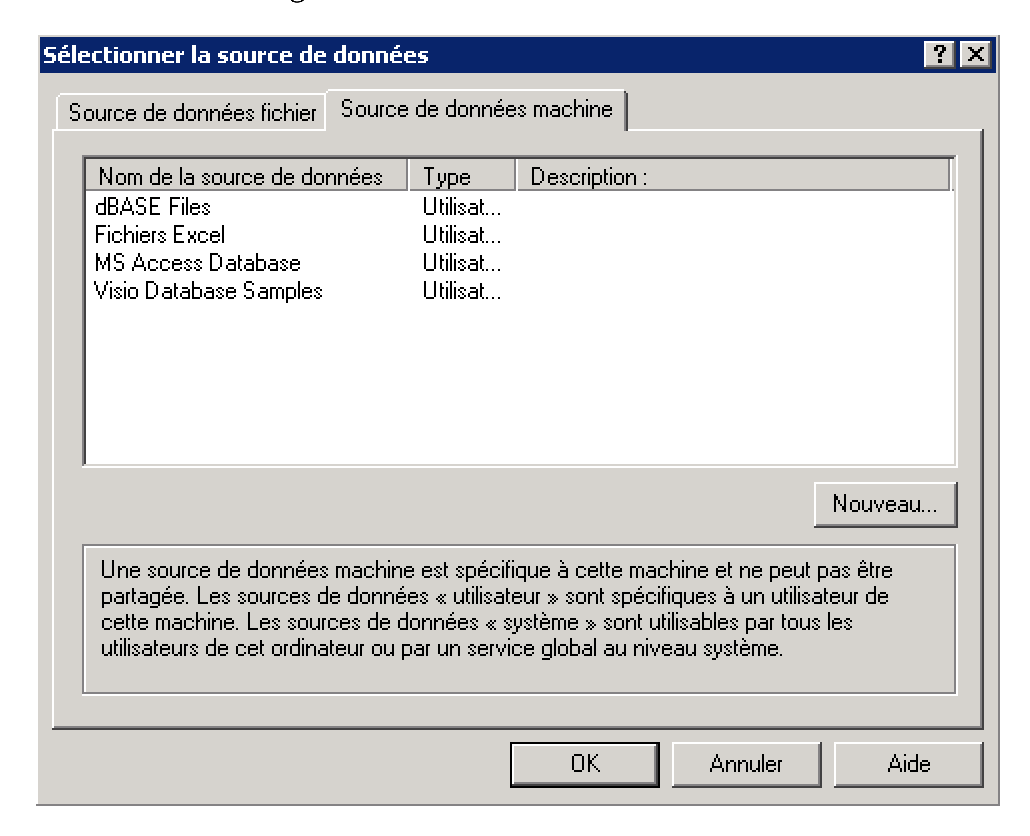
D'abord dans MS Access il faut aller dans *Fichier/Données Externes/Lier les tables...*:



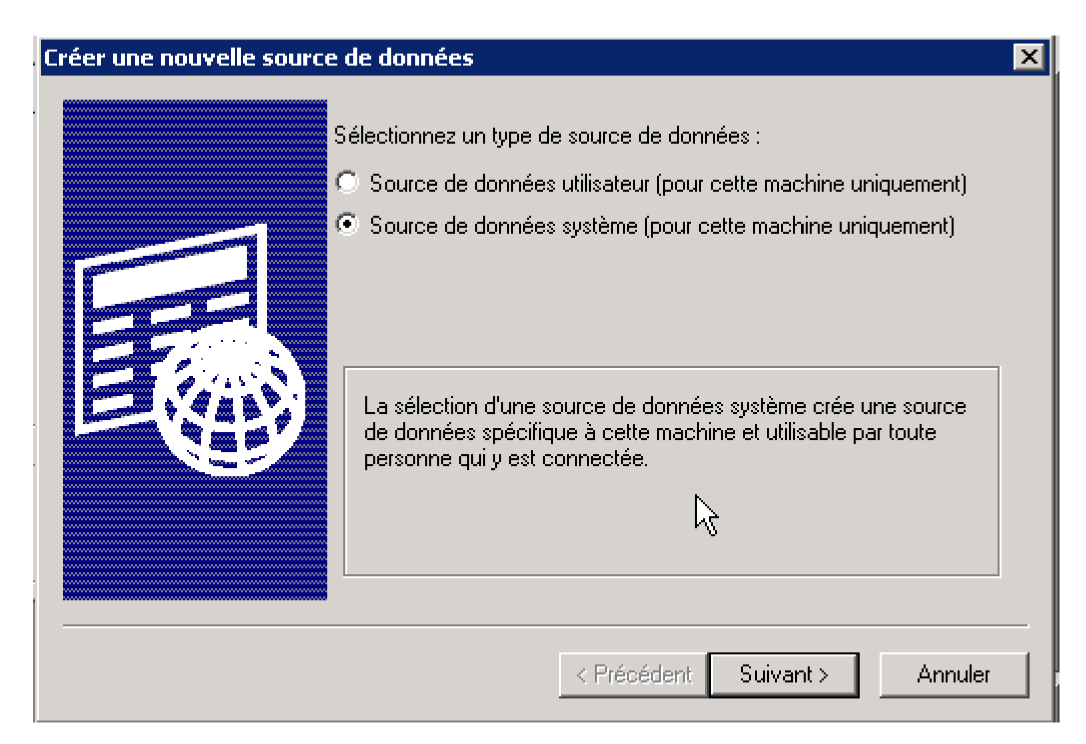
et choisir *ODBC*:



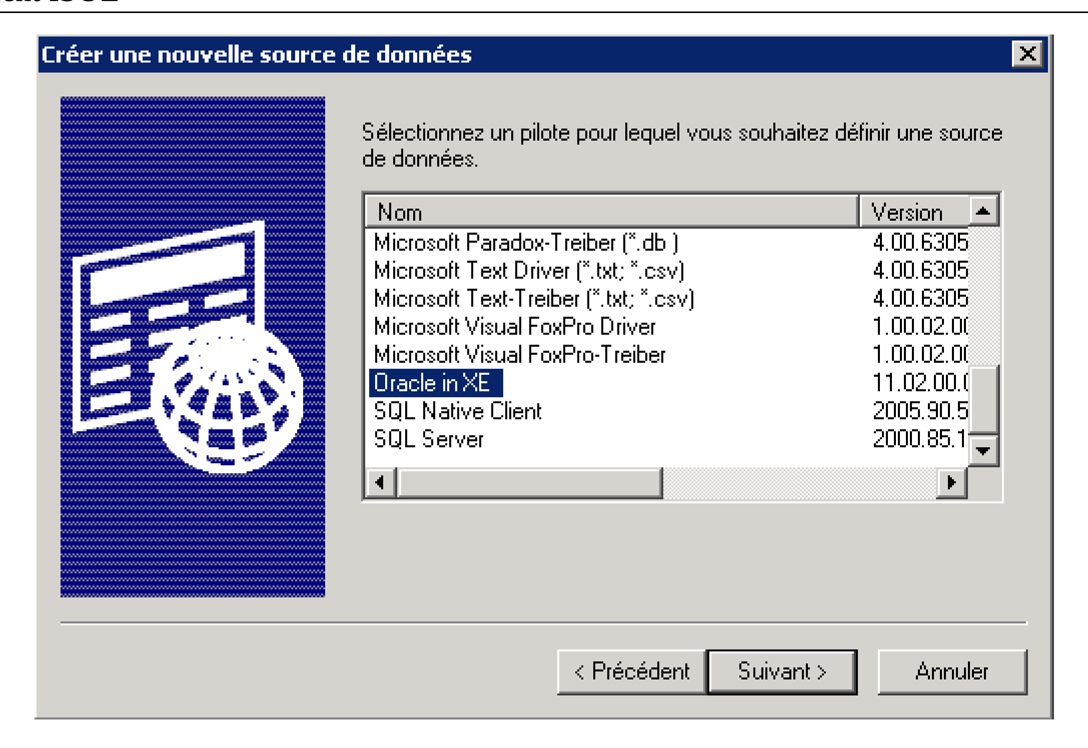
Ensuite dans la boîte de dialogue suivante:



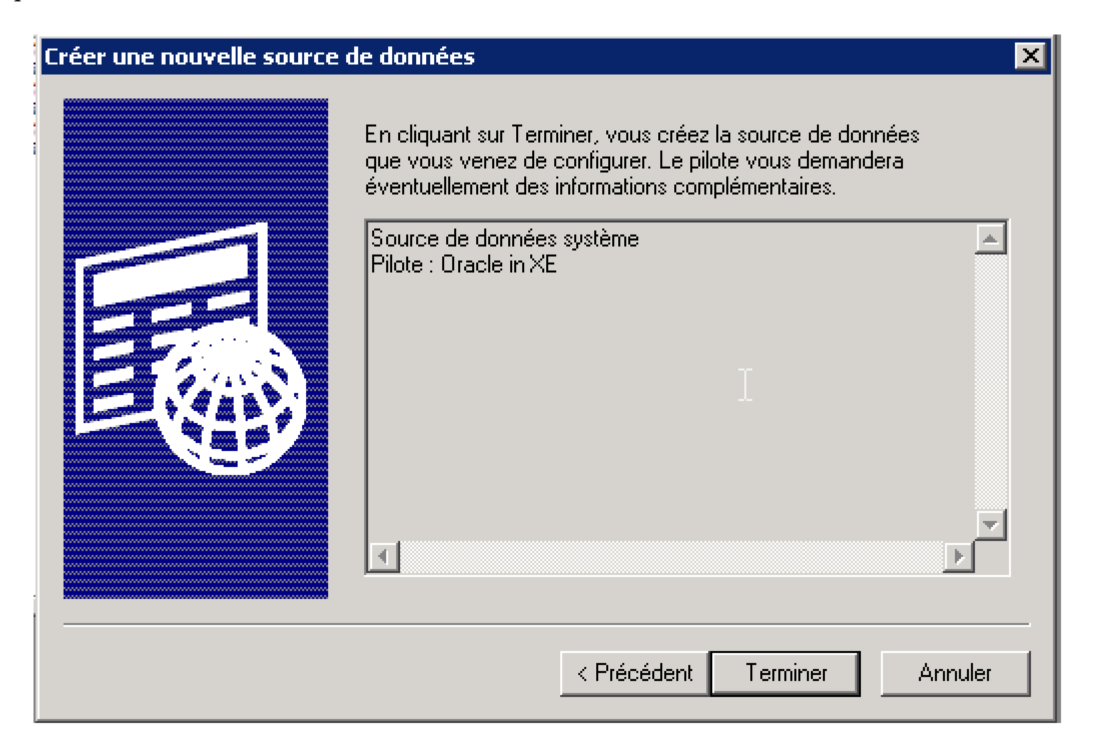
Cliquer sur *Nouveau...*:



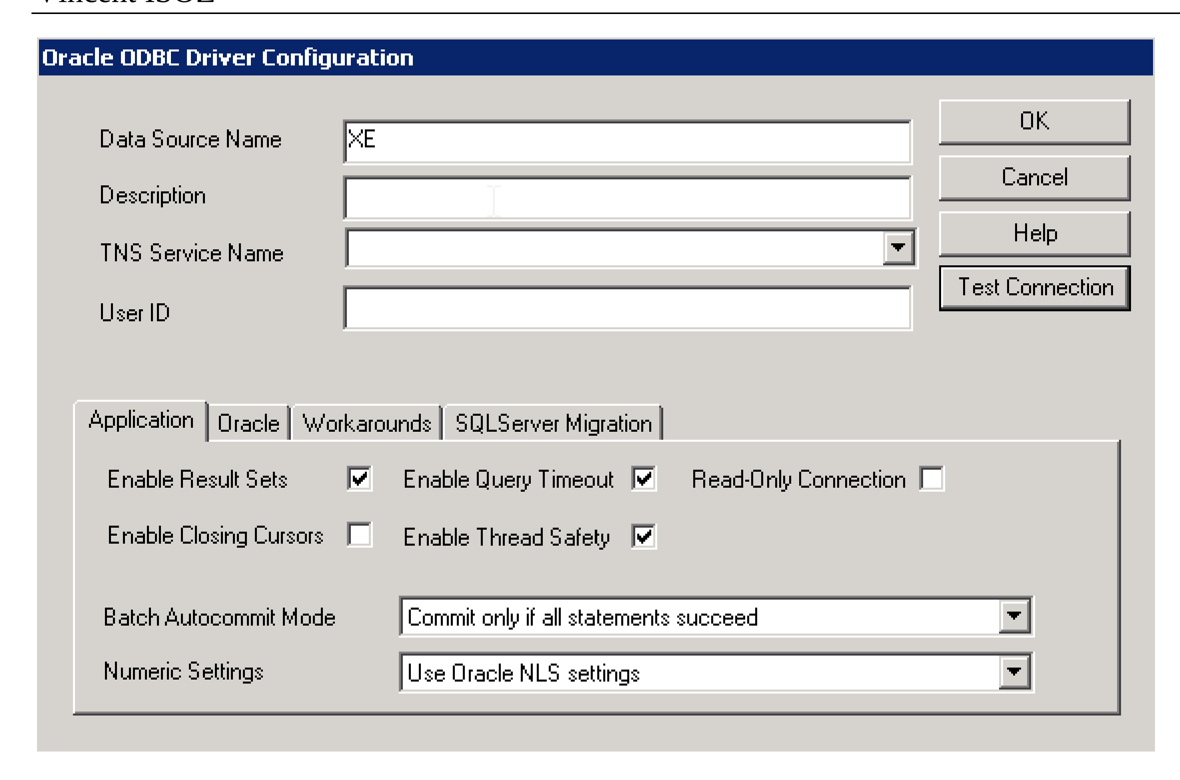
Prenez *Source de données système (pour cette machine uniquement)* Sélectionner le Driver *ODBC Oracle*:



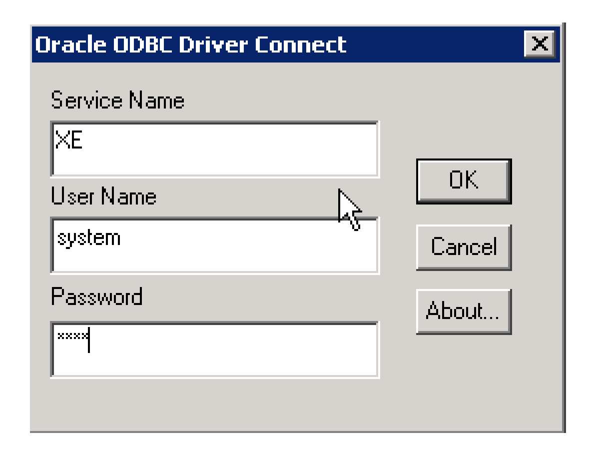
et cliquer sur *Suivant*:



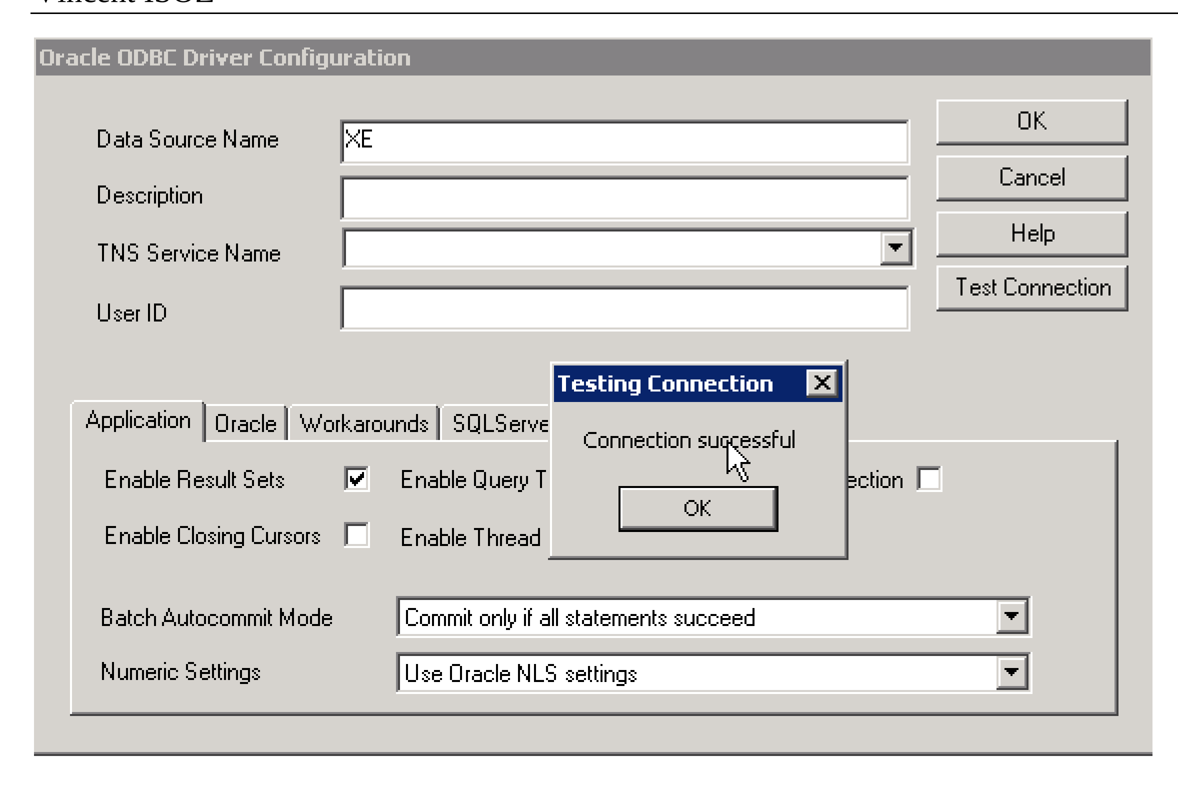
Et cliquez sur *Terminer*. Il vient alors:



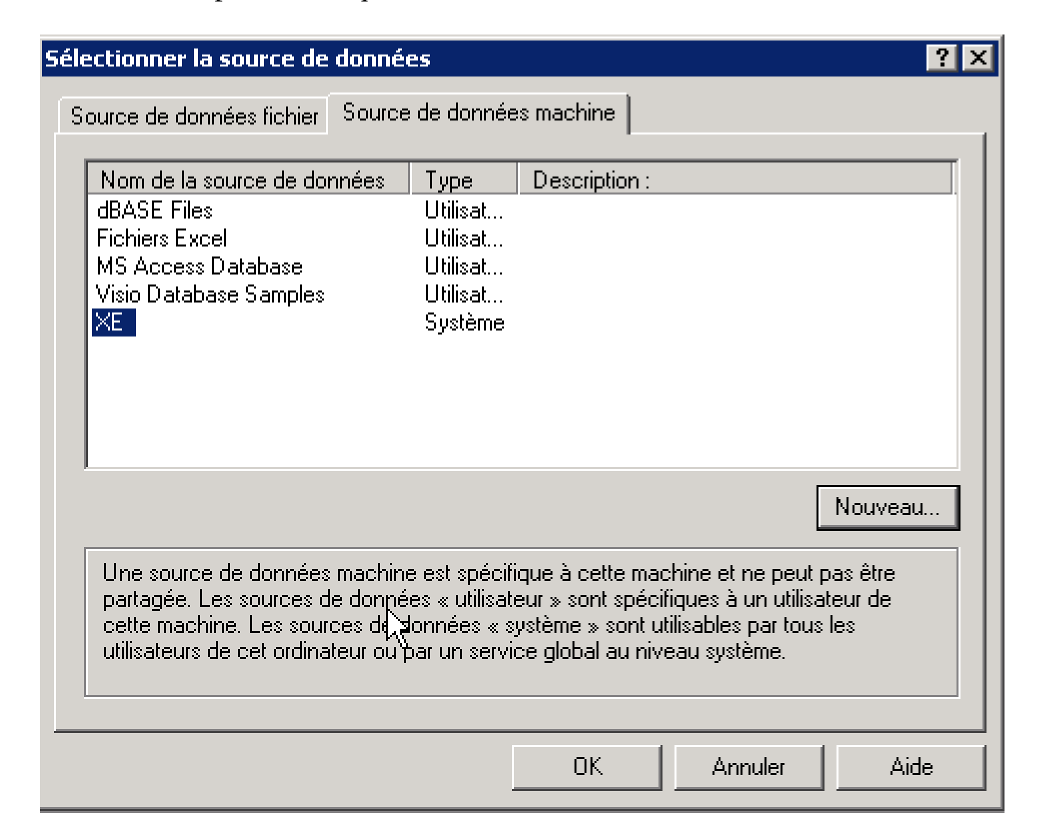
Nous cliquons sur *Test Connection* pour avoir la boîte de dialogue suivante (avec Oracle Express le nom du service est presque toujours *XE*):



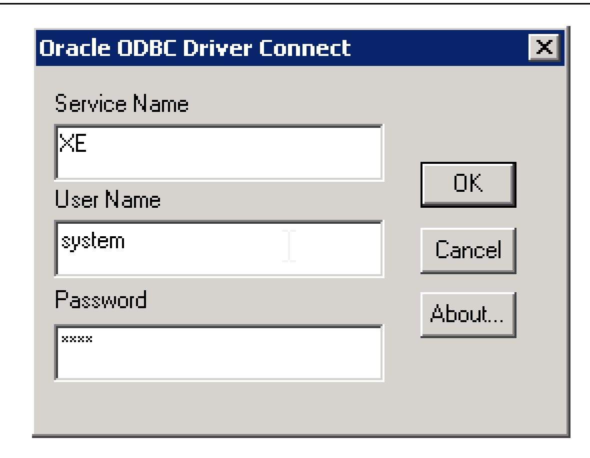
Sin nous cliquons sur *OK* il vient alors si les informations ci-dessus ont été correctement saisies:



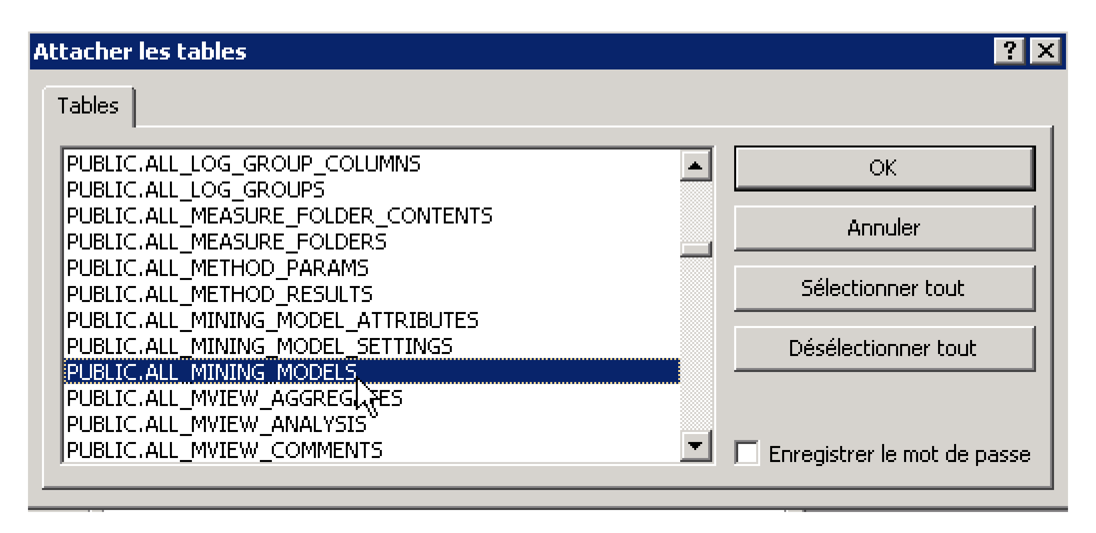
Ensuite, nous validons par *OK* ce qui nous ramène à:



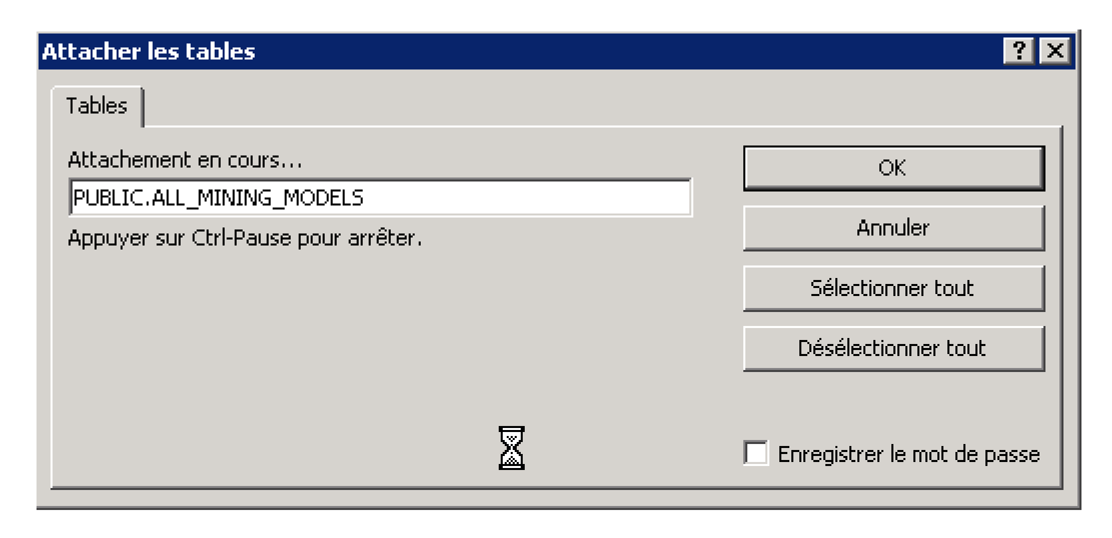
Si nous validons par *OK* il se peut que les informations suivante vous soit redemandées:



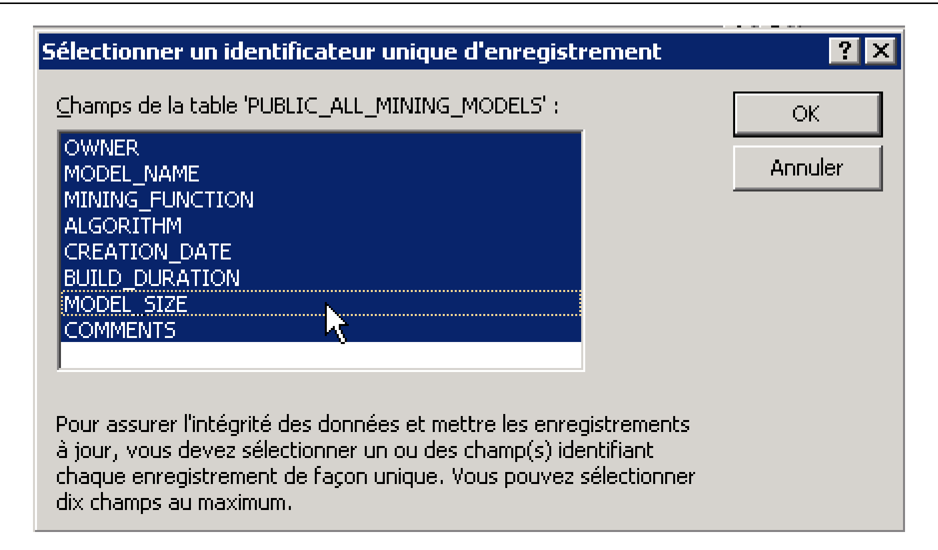
La liste des tables de la base apparaît alors à l'écran:



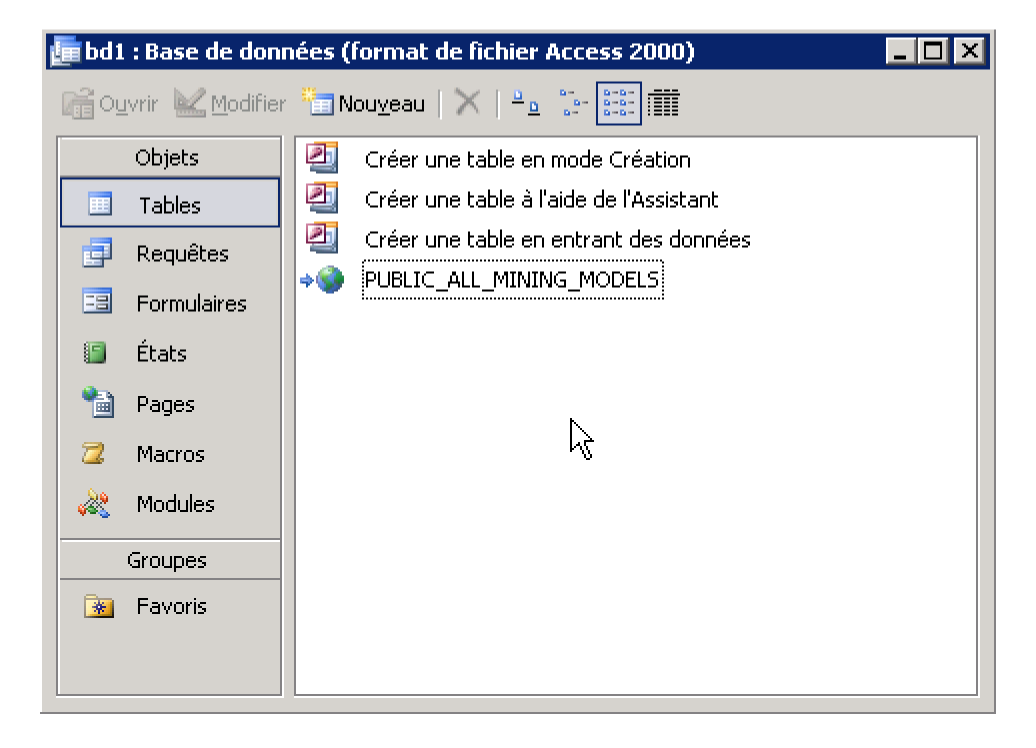
Si nous sélectionnons un table et validons par *OK* nous avons alors la table qui va se retrouver liée dans Microsoft Access après un petit moment d'attente:



et après avoir demandé la liste des champs de la table à importer:



Ce qui donnera au final une nouvelle table liée:



30 MySQL

Toujours dans le cadre de l'utilisation de MS Access en tant que base frontale, passons à la connexion de plus en plus courante avec MySQL (souvent utilisé comme SGBRD web).

Remarque: Le texte ci-dessous a été entièrement copier/coller du site http://blogmotion.fr

Pour permettre l'accès à la base MySQL de façon transparente au travers d'une application Microsoft nous devons installer un connecteur ODBC.

MySQL propose justement de télécharger ce connecteur MySQL ODBC gratuitement:

http://dev.mysql.com/downloads/connector/odbc

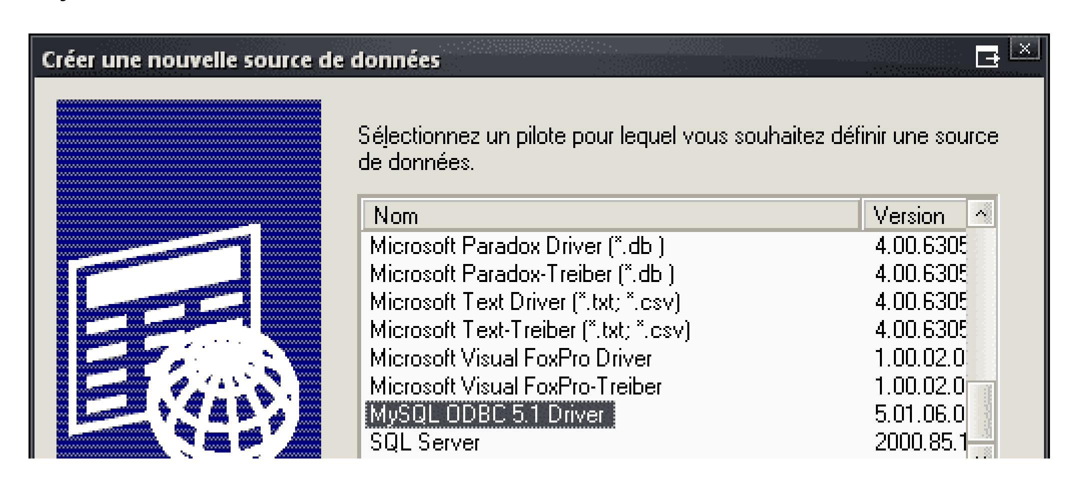
Installez-le!

Ensuite, rendez-vous dans le menu *Démarrer/Outils d'administration/Source de données (ODBC)*, ou via *Démarrer/Exécuter* et tapez *odbcad32.exe*

Cliquer sur *Ajouter*:

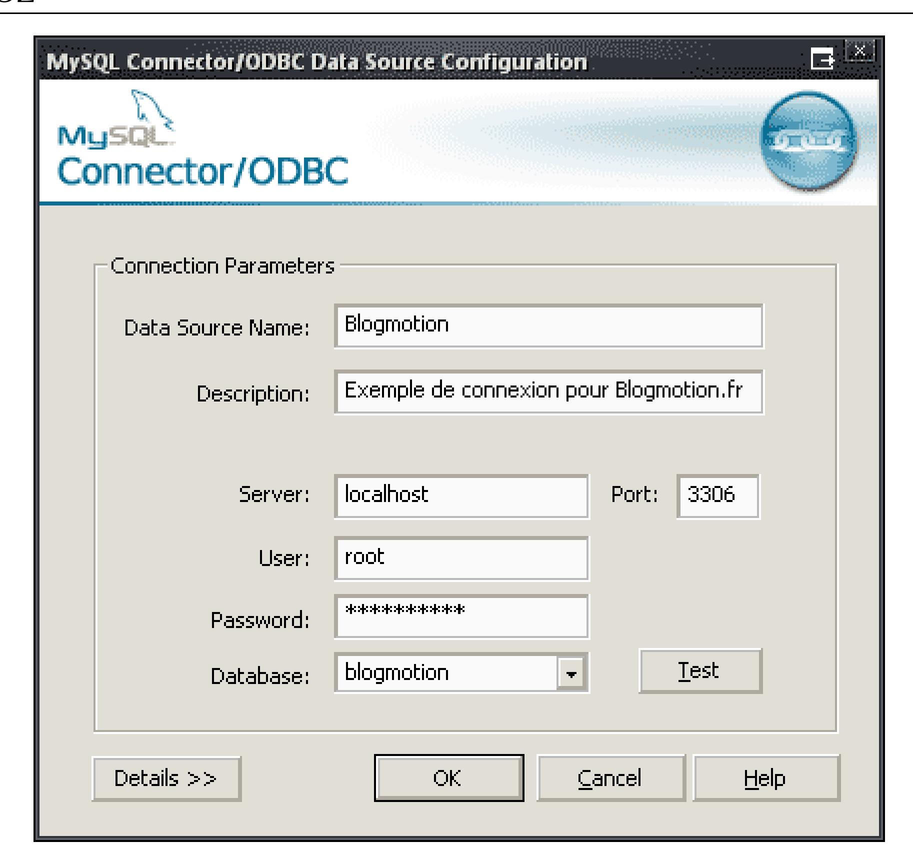


Choisir *MySQL ODBC*:



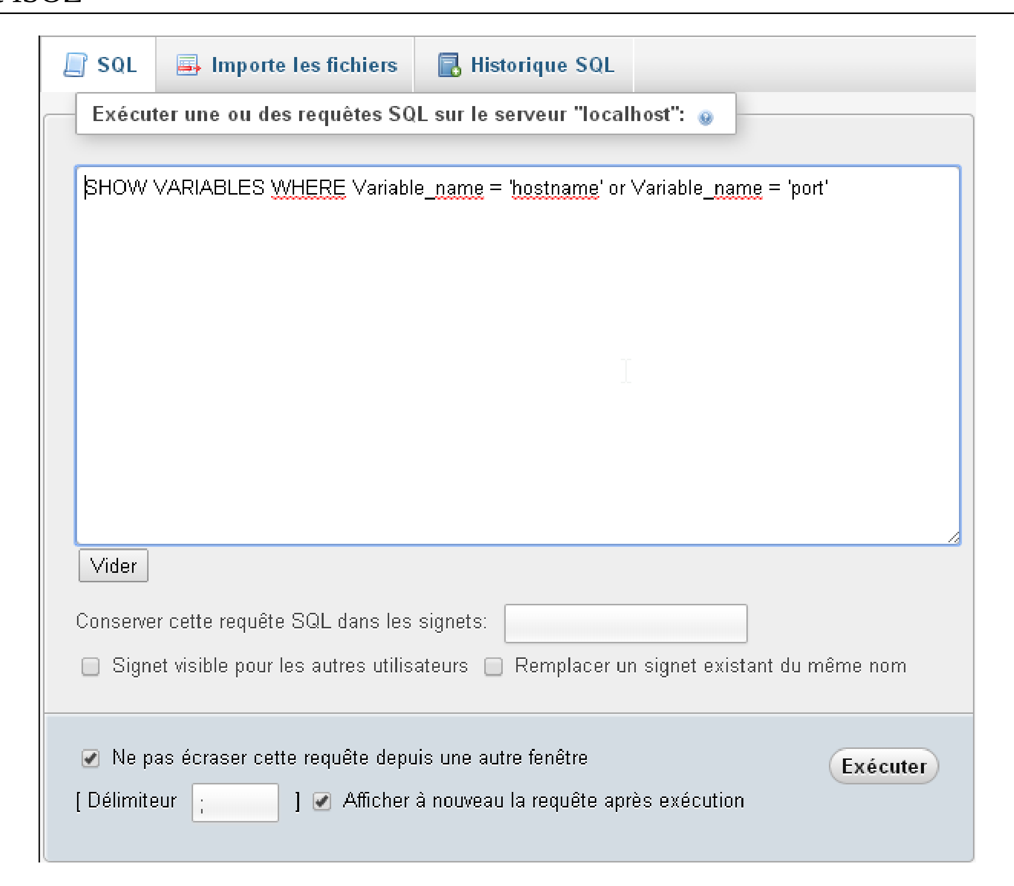
Puis cliquer sur *Terminer*.

Préciser un nom et une description pour ce connecteur (de votre choix) puis entrez le nom ou adresse IP du serveur MySQL (localhost pour moi) ainsi que l'utilisateur ayant accès à votre base de données:

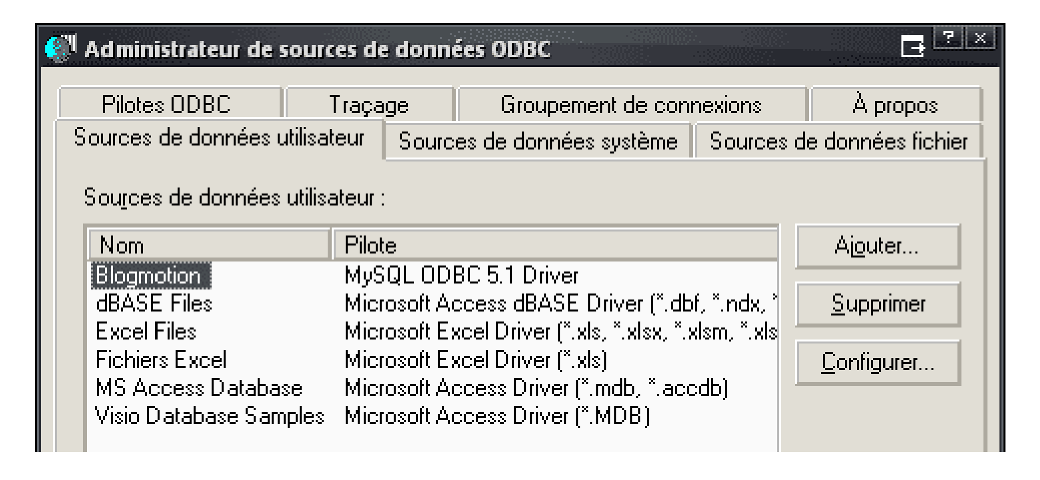


Vous devrez obtenir *Connection Successful* si vous cliquez sur le bouton *Test*.

Si ce n'est pas le cas, pensez à vérifier que votre serveur MySQL fonctionne et que celui-ci est accessible à partir de l'extérieur, l'accès est parfois restreint sur l'interface de boucle local (localhost) 127.0.0.1. Si vous ne connaissez pas le nom ou le port de votre serveur MySQL vous pouvez écrir les deux commandes suivante dans le système de requête de MySQL pour obtenir ces deux informations:

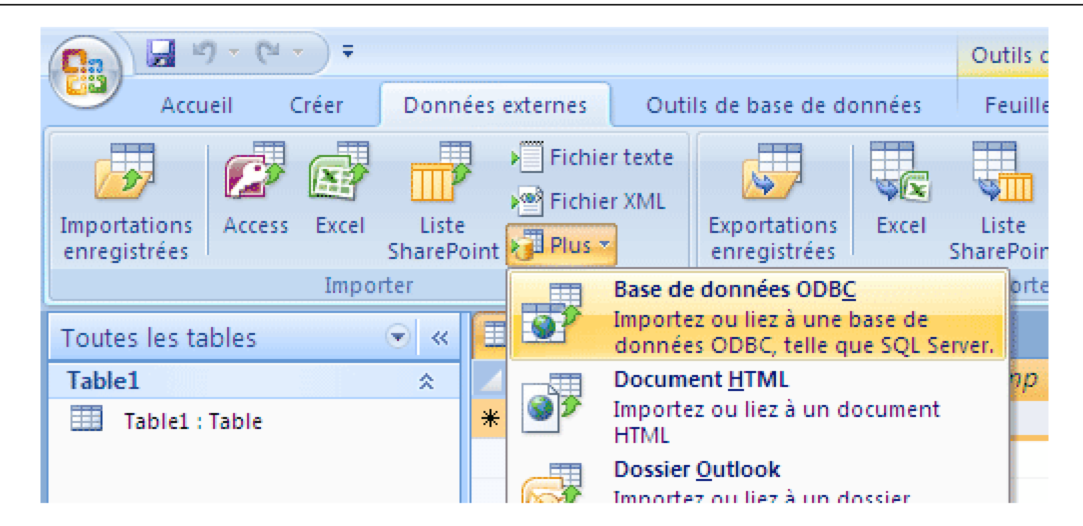


Cliquer sur *OK*, vous devriez obtenir ceci (excepté le nom…):

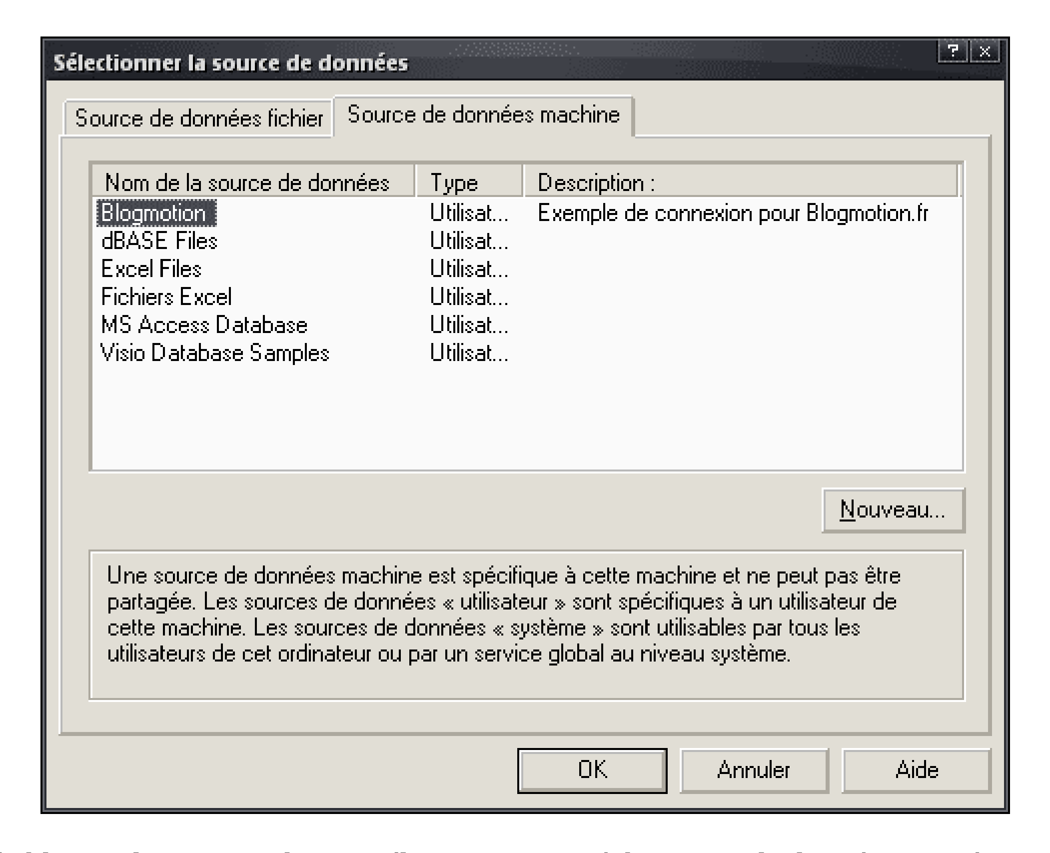


C'est terminé pour la partie ODBC.

Maintenant, dans Access 2007 ou ultérieur allez dans l'onglet *Données externes*/Plus/*Base de données ODBC*:



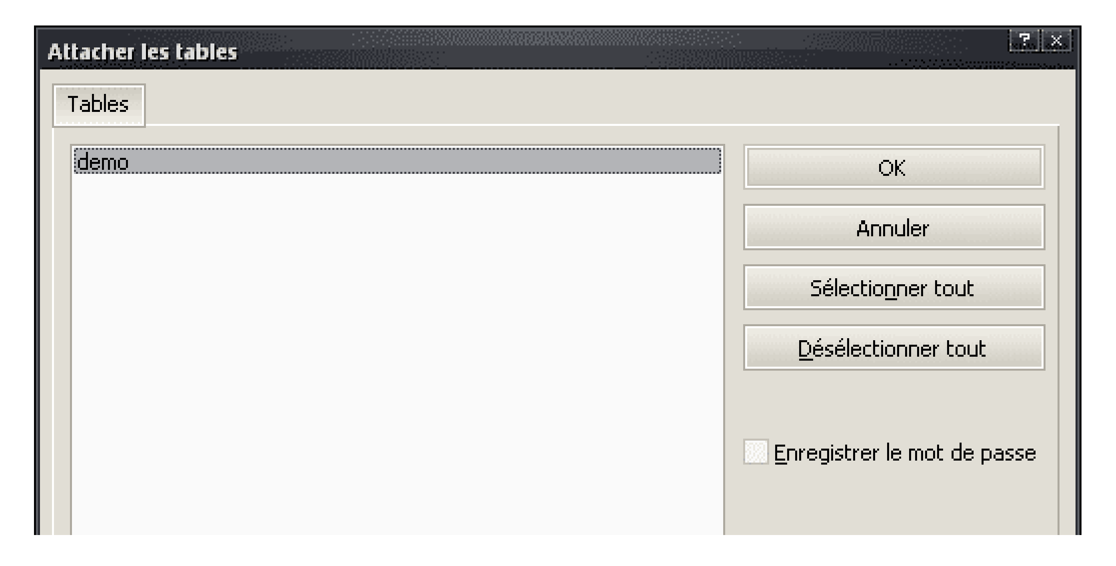
Dans le deuxième onglet *Source de données machine* sélectionner le Connecteur précédemment créé:



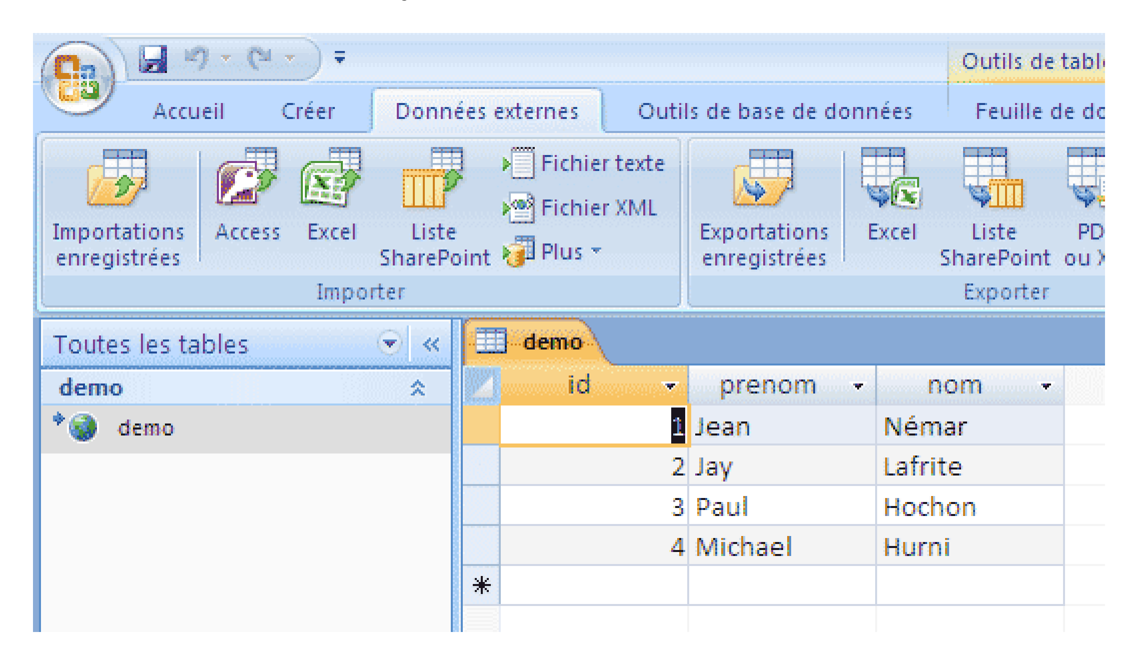
Pour établir une liaison synchrone, sélectionner *Lier à la source de données en créant une table attachée*:



Sélectionner que vous souhaitez exploiter et cliquer sur *OK*:



Vous retrouvez vos données de MySQL dans Access:

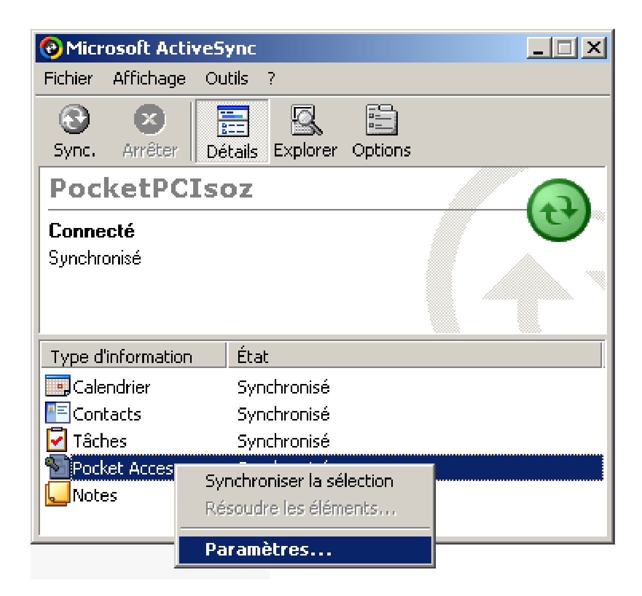


Attention: chaque modification effectuée sera répercutée sur votre base MySQL!

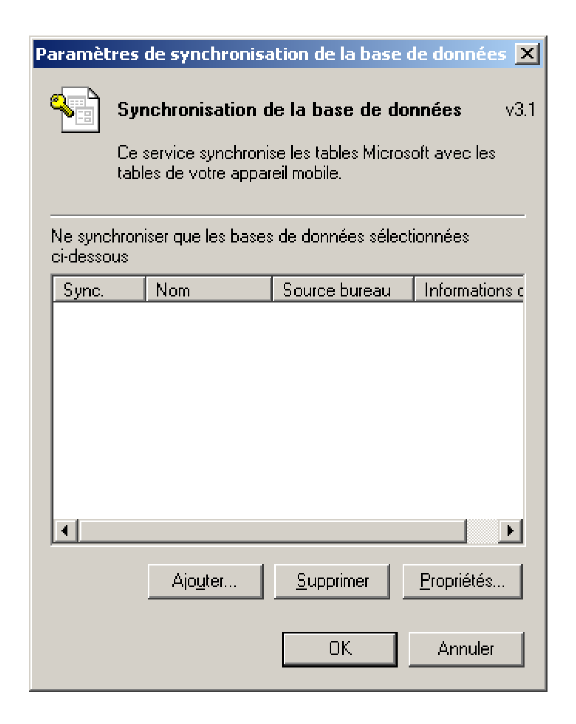
31 PocketPC

Voici très probablement le point le plus intéressant de ce support de cours: comment rendre votre base de données MDB mobile sur PocketPC.

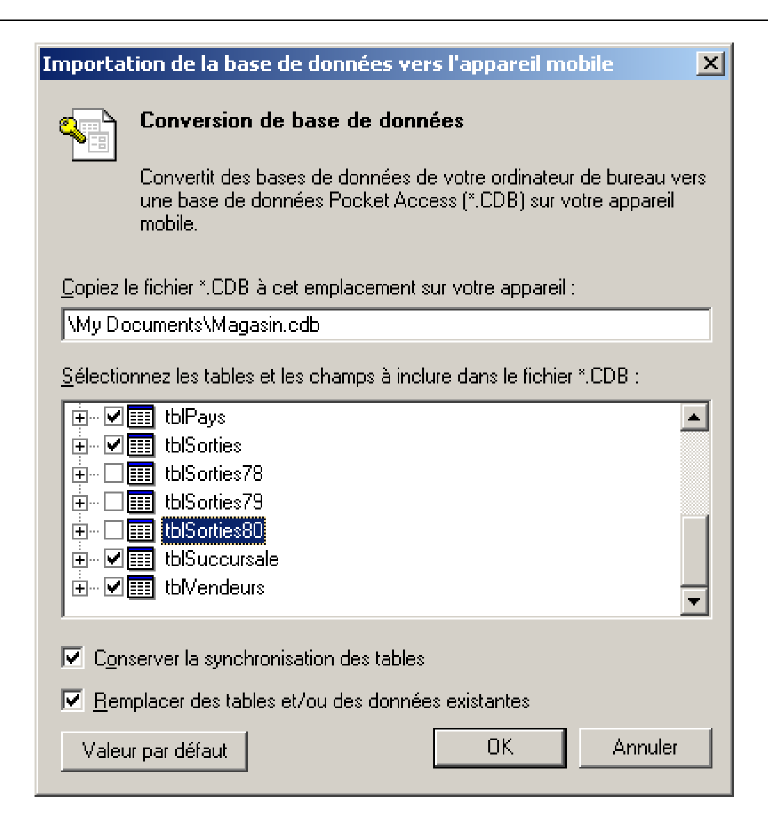
Nous allons synchroniser notre base *Magasin.mdb* située dans notre dossier *Mes documents* avec notre PocketPC. Pour cela, dans ActiveSync 3.7.1 voici comment effectuer l'opération:



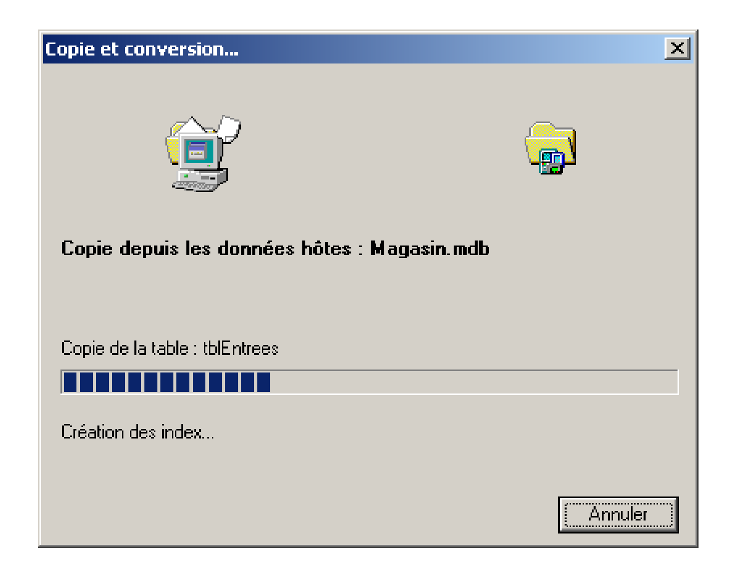
S'ouvre une boîte de dialogue vous demandant où se situe votre base. Il vous suffit de cliquer sur *Ajouter:*



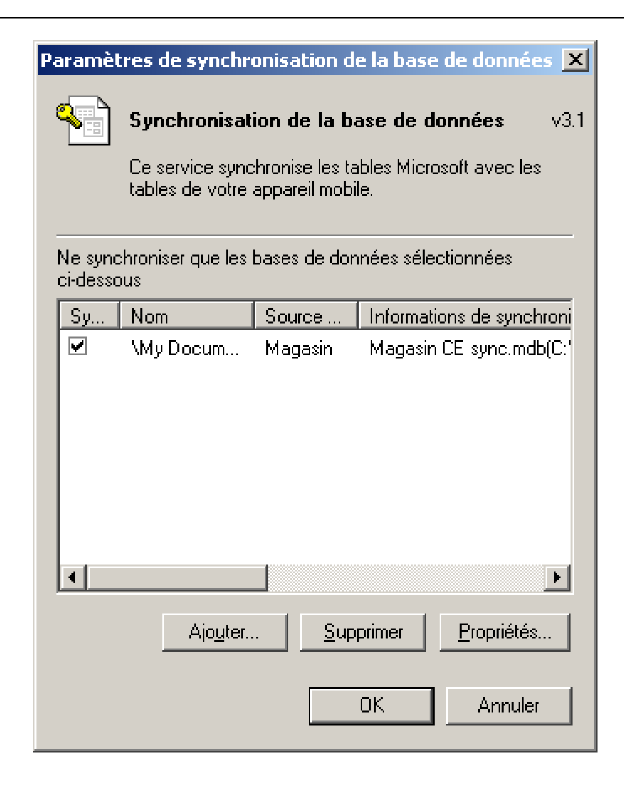
Sélectionnez les tables que vous souhaitez synchroniser:



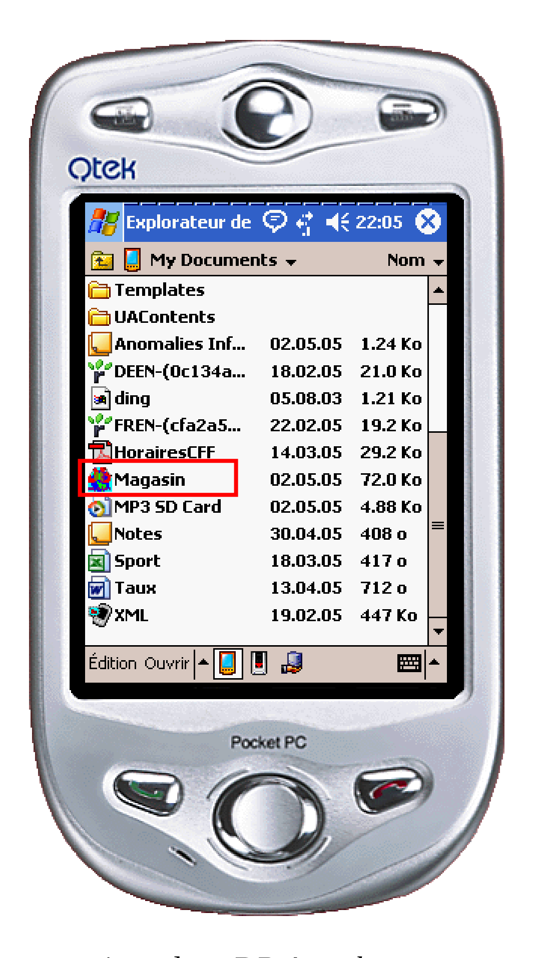
Après avoir cliqué sur *OK*, la conversion se fera:



et celle-ci une fois terminée:



et cliquez sur *OK*. Une fois ceci fait, en allant dans l'explorateur de votre PocketPC vous aurez:



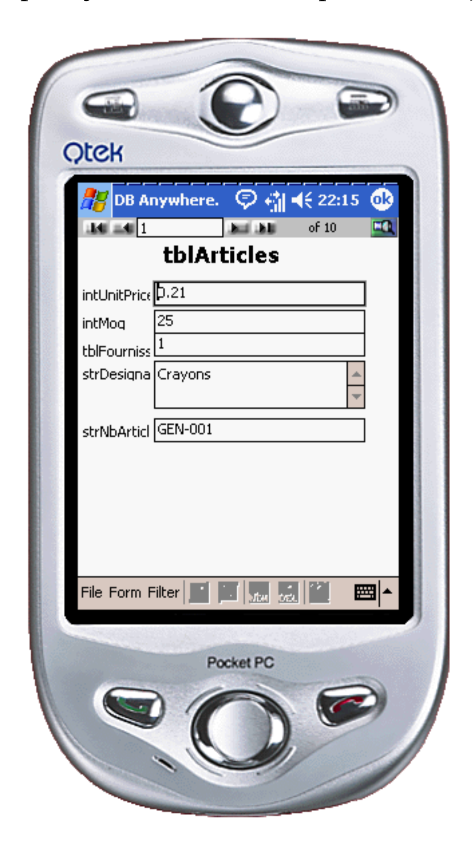
Ouvrant la base *Magasin*, vous arrivez dans *DB Anywhere*:





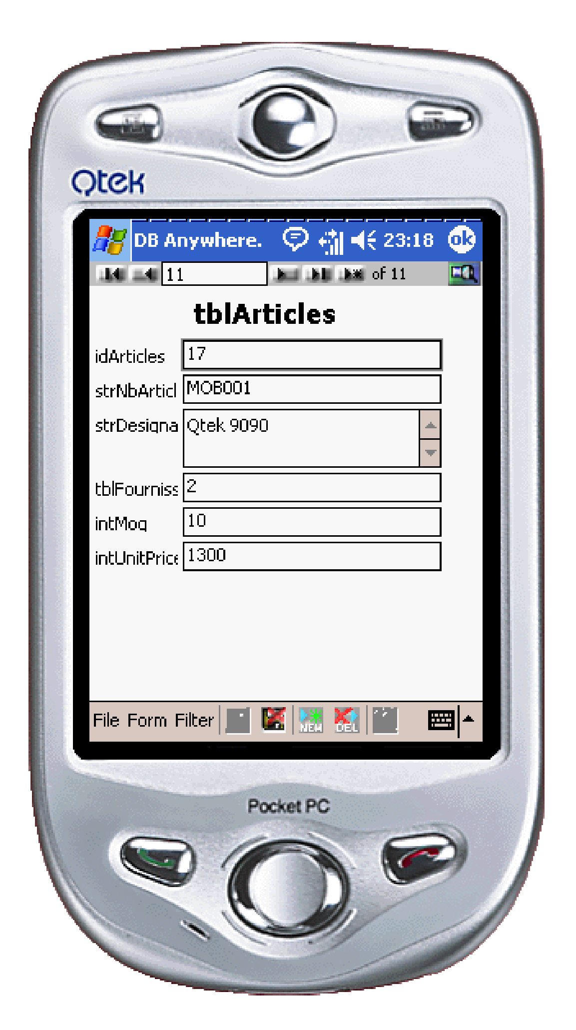
Nous retrouvons donc leurs tables avec leurs données respectives par contre, les requêtes et formulaires que nous avions créé ne sont plus disponibles.

Après avoir joué deux ou trois minutes avec DB Anywhere (qui vous permet quand même d'utiliser des filtres, requêtes, formulaires mais ne gère pas les relations entre les tables à ce jour) nous pouvons faire un petit formulaire (attention dans les options de le mettre en *Full Access* sinon vous ne pourrez pas ajouter de données par la suite!):

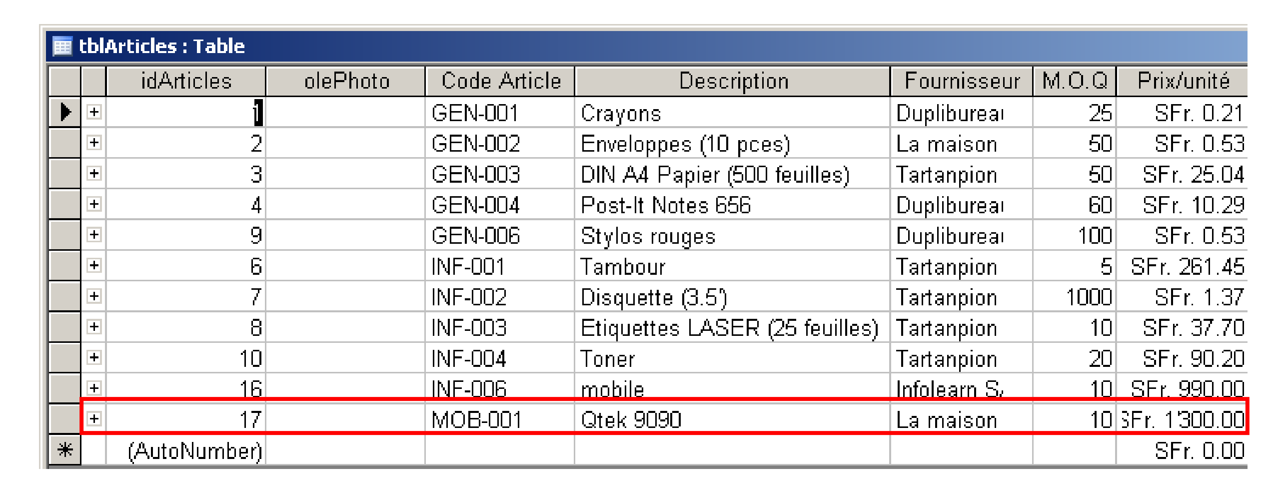


Faites attention aux clés primaires usant de champ de numération qui étaient automatiques dans MS Access car dans DB Anywhere cela n'existe pas et alors lors de la synchronisation vous aurez un message d'erreur.

L'ajout d'une nouvel article (n'oubliez pas de l'enregistrer en cliquant sur la petite disquette):



Après synchronisation avec ActiveSync nous retrouvons bien dans MS Access:



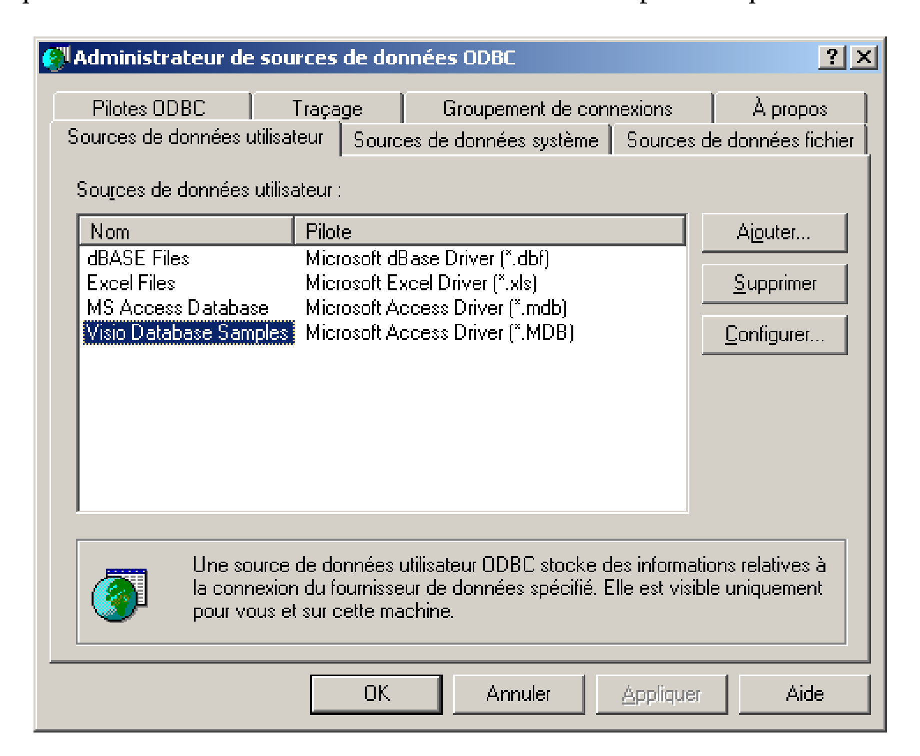
tblArticles : Table

idArticles	olePhoto	Code Article	Description	Fournisseur	M.O.Q	Prix/unité
1		GEN-001	Crayons	Duplibureau	25	SFr. 0.21
2		GEN-002	Enveloppes (10 pces)	La maison	50	SFr. 0.53
3		GEN-003	DIN A4 Papier (500 feuilles)	Tartanpion	50	SFr. 25.04
4		GEN-004	Post-It Notes 656	Duplibureau	60	SFr. 10.29
9		GEN-006	Stylos rouges	Duplibureau	100	SFr. 0.53
6		INF-001	Tambour	Tartanpion	5	SFr. 261.45
7		INF-002	Disquette (3.5')	Tartanpion	1000	SFr. 1.37
8		INF-003	Etiquettes LASER (25 feuilles)	Tartanpion	10	SFr. 37.70
10		INF-004	Toner	Tartanpion	20	SFr. 90.20
16		INF-006	mobile	Infolearn S.	10	SFr. 990.00
17		MOB-001	Qtek 9090	La maison	10	SFr. 1'300.00
(AutoNumber)						SFr. 0.00

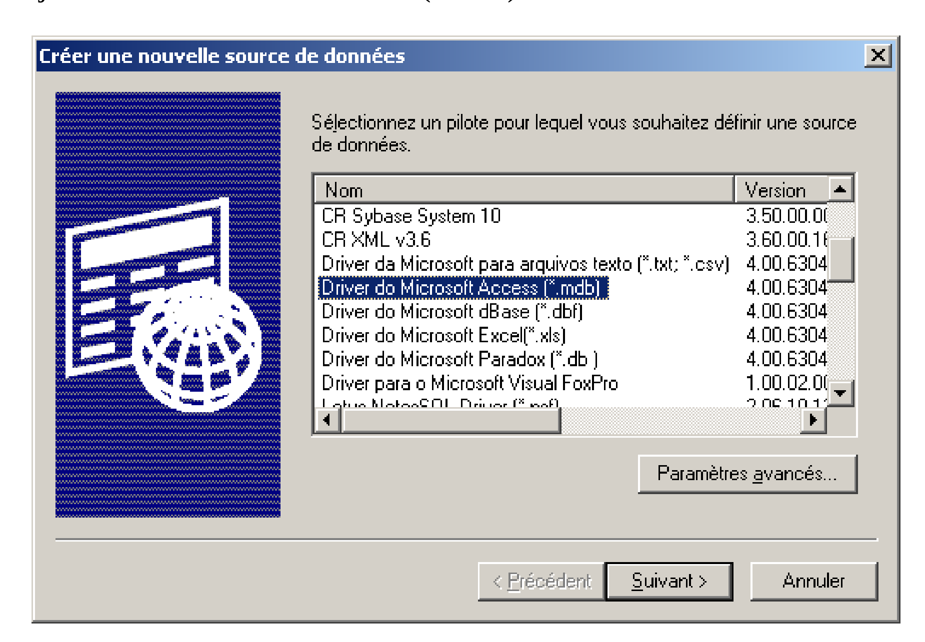
32 Business Objects

Nous allons observer dans ce chapitre comment connecter une base de données MS Access au logiciel BusinessObjects 5.0

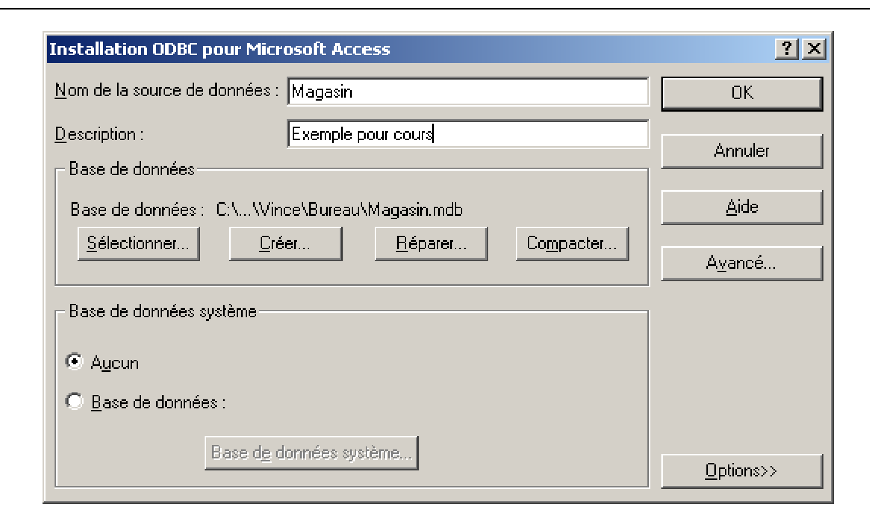
Nous allons donc continuer avec notre base de données *Magasin.mdb*. Dans les outils d'administration de MS Windows allez dans *Source de données (ODBC)* nous en aurons besoin pour que BO se connecte à notre base et suivez les étapes indiquées ci-dessous:



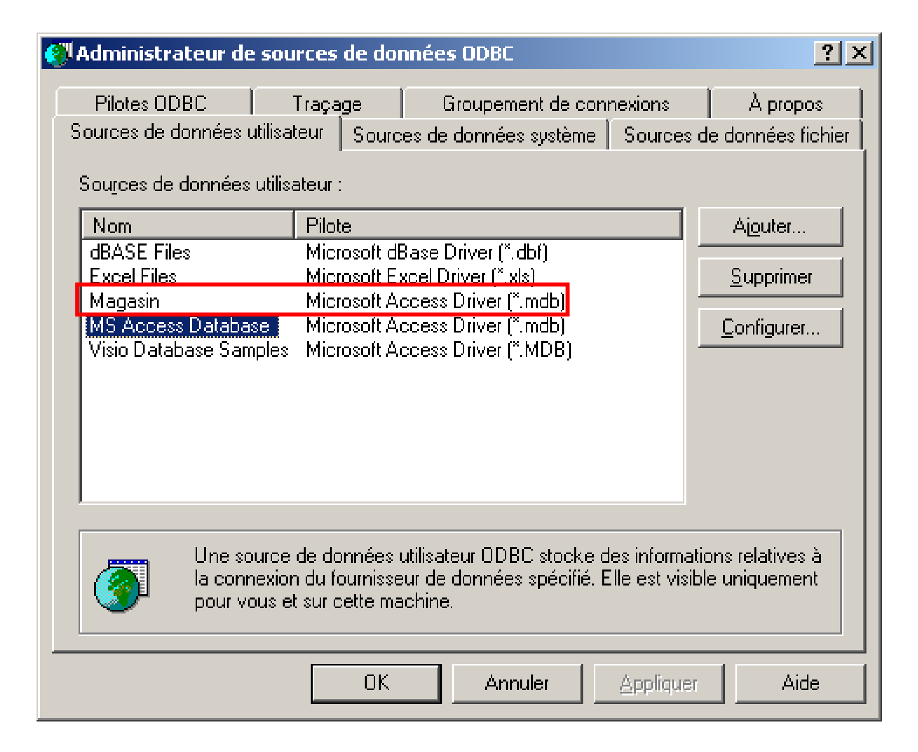
Cliquez sur Ajouter et sélection une source (driver) de données MS Access:



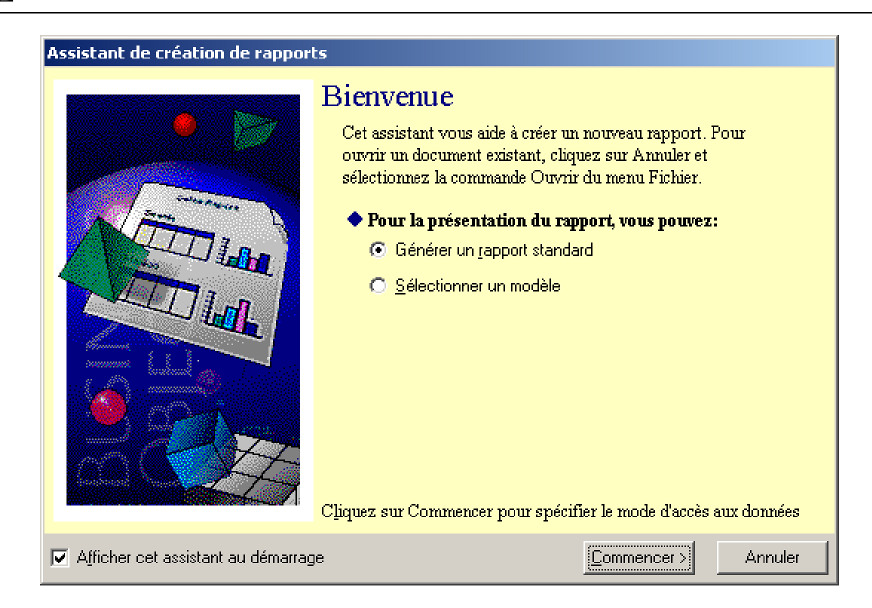
cliquez sur *Suivant* et allez cherchez la base de données Magasin.mdb:



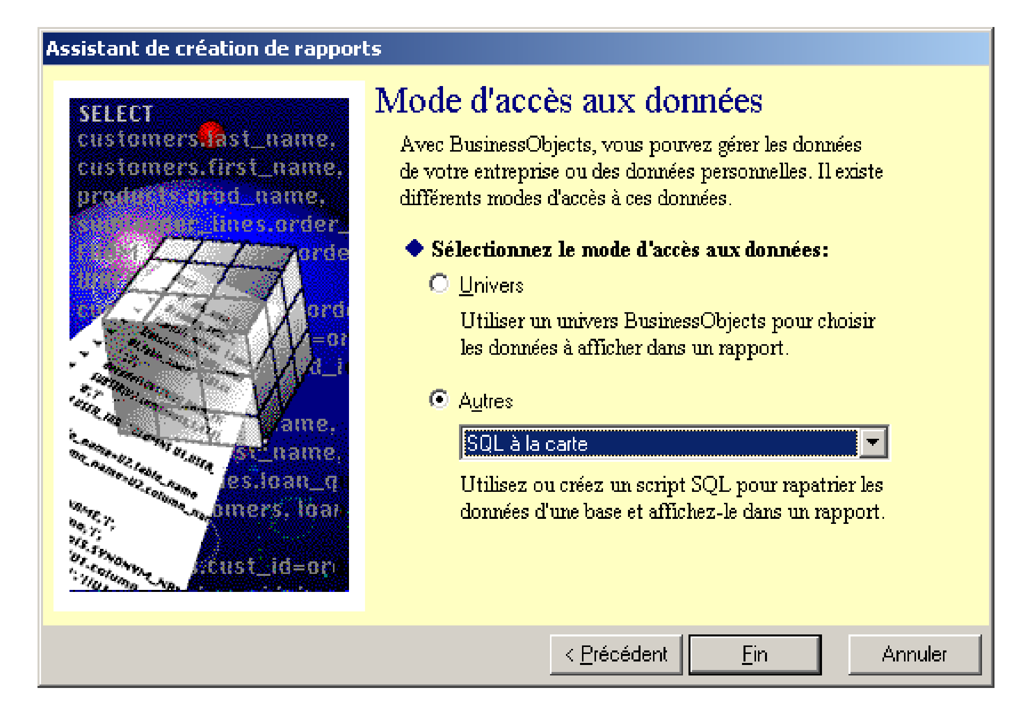
Quand vous cliquerez sur OK, vous aurez:



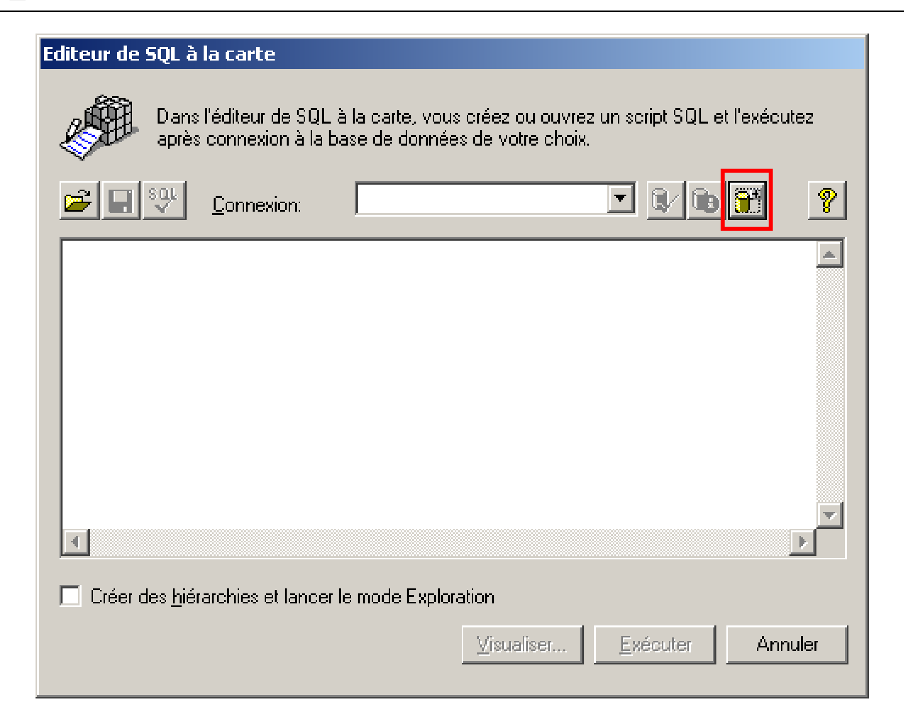
validez par OK et quittez ODBC pour lancer BO:



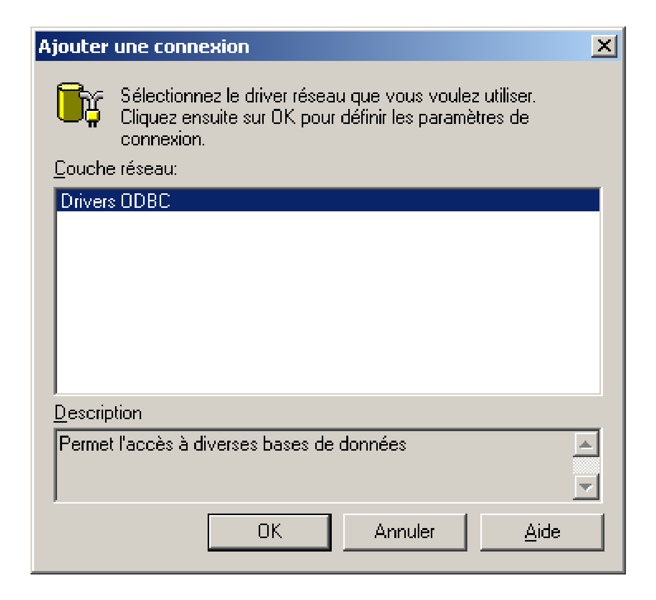
Sélectionnez la première option et lorsque vous cliquez sur *Commencer* choisissez SQL à la carte:



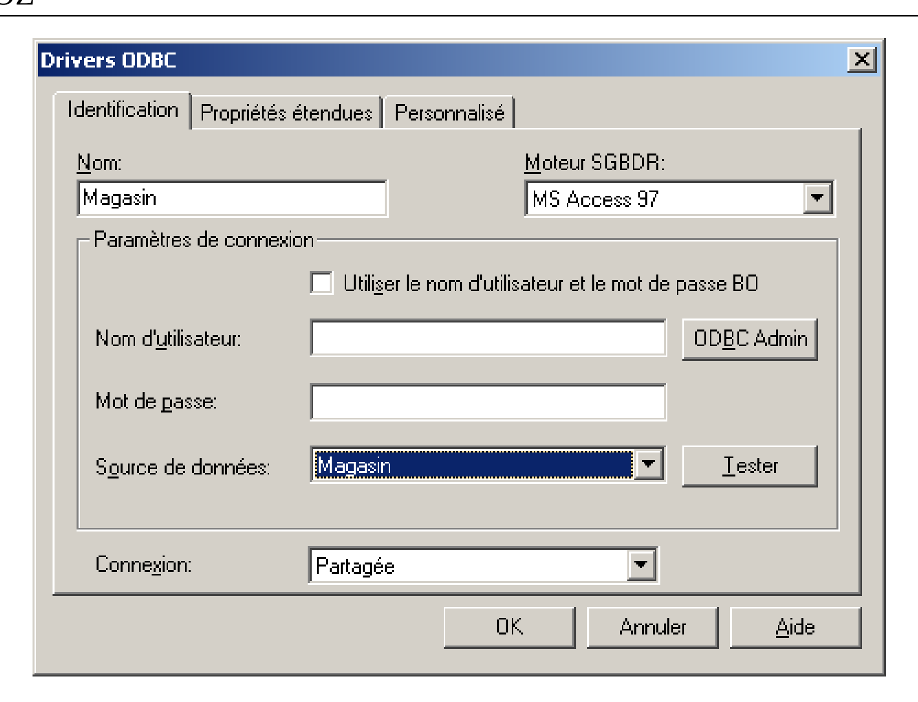
Dans la fenêtre suivante, cliquez sur le bouton indiqué ci-dessous par aller cherche la source de données utilisateur *Magasin.mdb*:



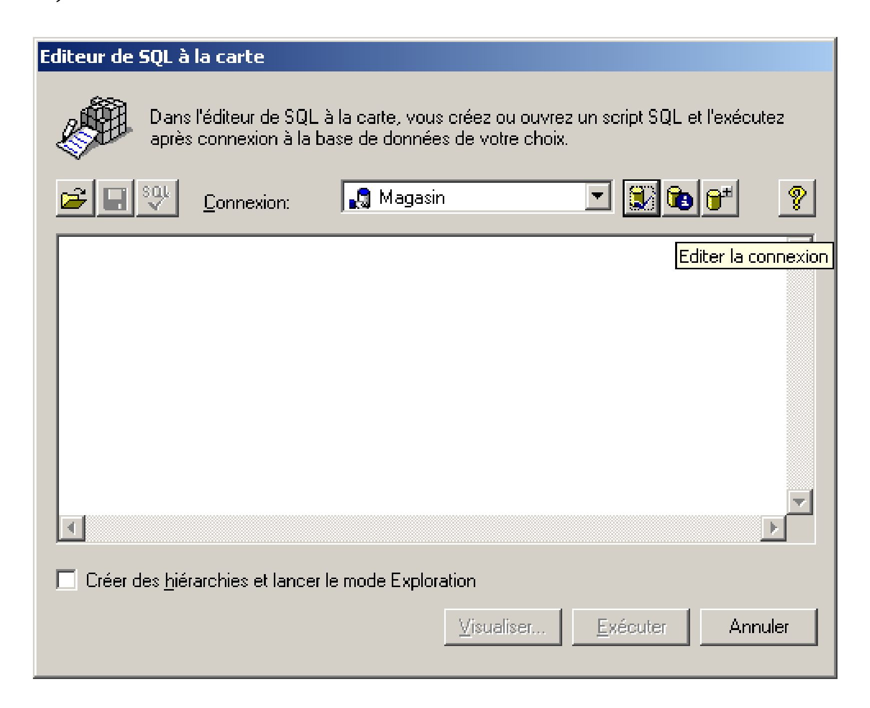
et sélectionnez ODBC sur la fenêtre qui apparaît:



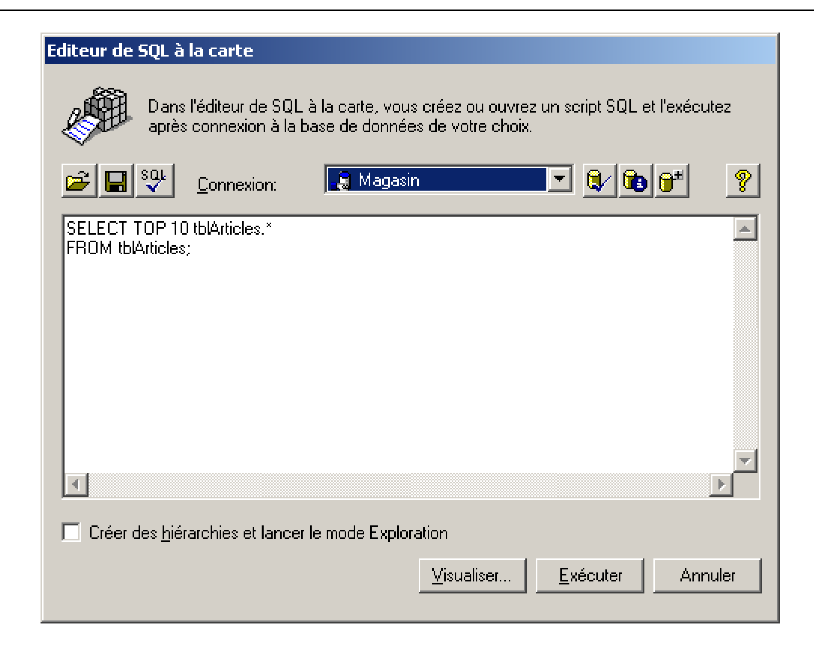
Ensuite, prenez garde à bien sélectionne *Magasin* comme source de données:



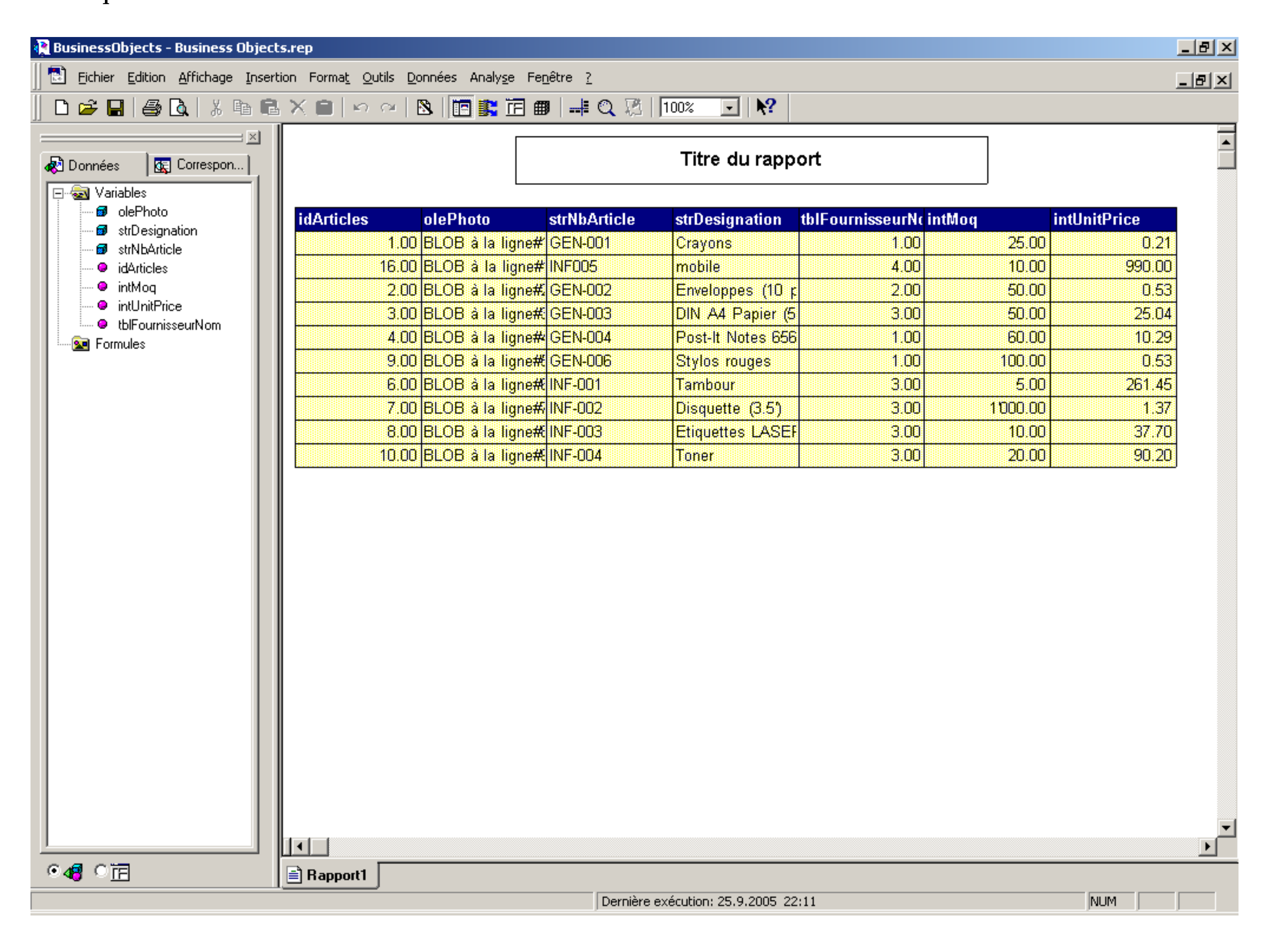
et validez par *OK*, vous aurez alors:



écrivez votre requête SQL (exemple pris du cours MS Access plus haut):



et cliquez sur *Exécuter*. Vous aurez alors:

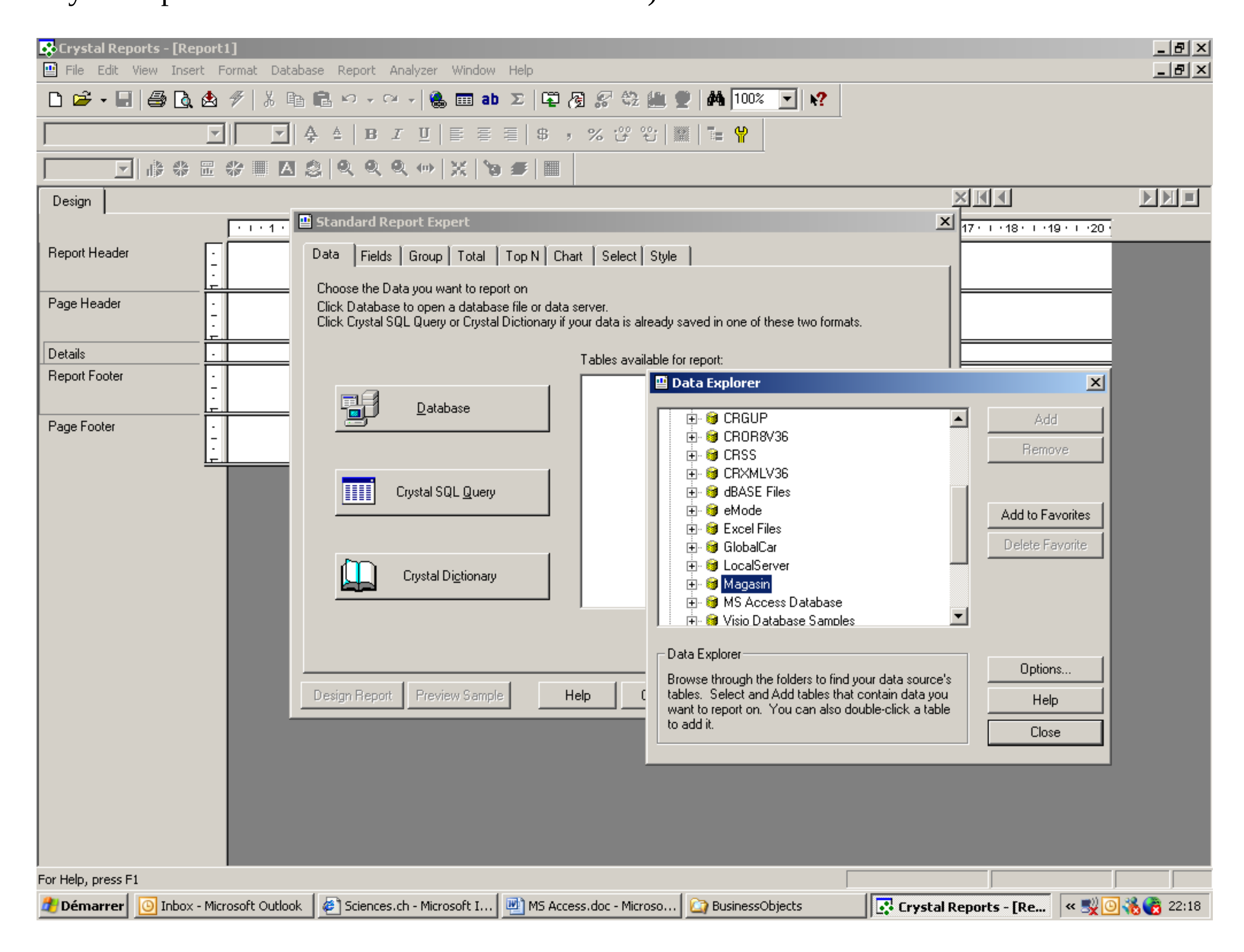


idArticles	olePhoto	strNbArticle	strDesignation	tblFournisseurN	intMoq	intUnitPrice
1.00	BLOB à la ligne#	GEN-001	Crayons	1.00	25.00	0.21
16.00	BLOB à la ligne#	INF005	mobile	4.00	10.00	990.00
2.00	BLOB à la ligne#	GEN-002	Enveloppes (10 p	2.00	50.00	0.53
3.00	BLOB à la ligne#	GEN-003	DIN A4 Papier (5	3.00	50.00	25.04
4.00	BLOB à la ligne#	GEN-004	Post-It Notes 656	1.00	60.00	10.29
9.00	BLOB à la ligne#	GEN-006	Stylos rouges	1.00	100.00	0.53
6.00	BLOB à la ligne#	INF-001	Tambour	3.00	5.00	261.45
7.00	BLOB à la ligne#	INF-002	Disquette (3.5')	3.00	1000.00	1.37
8.00	BLOB à la ligne#	INF-003	Etiquettes LASEF	3.00	10.00	37.70
10.00	BLOB à la ligne#	INF-004	Toner	3.00	20.00	90.20

Ensuite c'est du "bête" usage de BusinessObjects (concept d'utilisation similaire à MS Excel avec la possibilité d'ajouter des colonnes dans le tableau avec des formules, filtres, tris, tableau croisé, etc.).

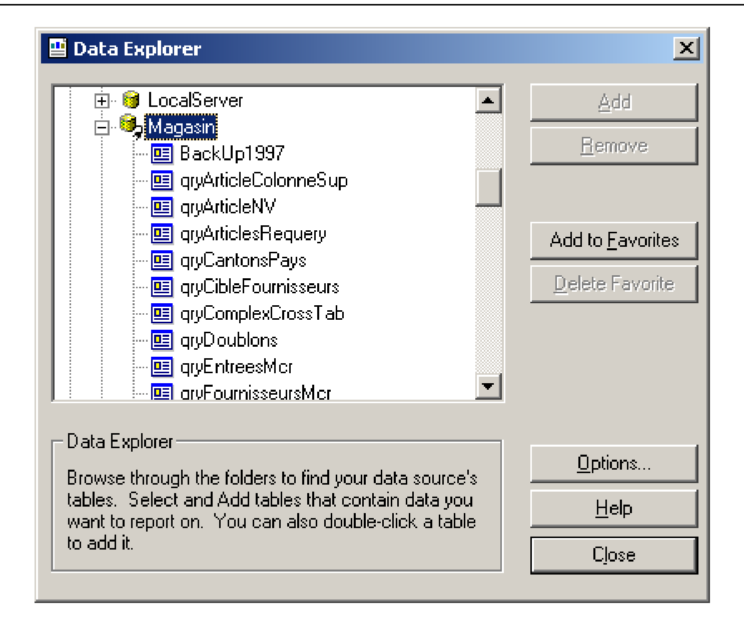
33 Crystal Reports

L'utilisation de Crystal Reports pour notre base *Magasin.mdb* tient à nouveau par les mêmes procédures initiales de création d'un driver ODBC nommé *Magasin* comme montré ci-dessous (assistant obtenu en cliquant sur le bouton *Nouveau* en haut à gauche de l'écran de CrystalReports et ensuite sur le bouton *Database*).

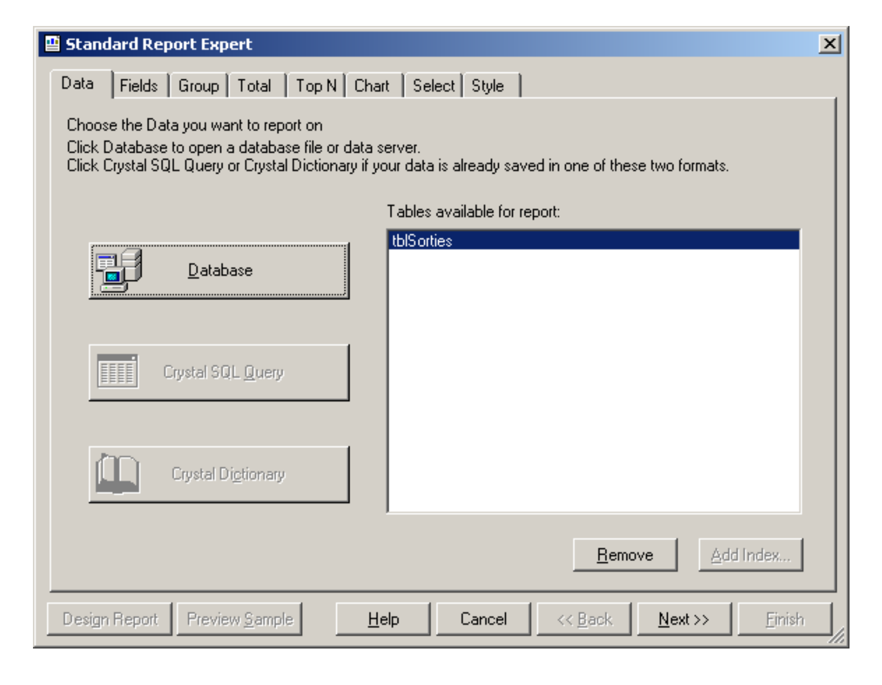


Remarque: un pro passera par *Crystal SQL Query*

Avec un double clic sur *Magasin*, vous obtiendrez:



où l'on voit aussi bien les tables que les requêtes ! En faisant un double clic sur la table *tblSorties*, vous obtiendrez:



Ensuite, il vous suffit de parcourir les onglets *Fields*, *Group*, *Total*, *Top N*, *Chart*, *Select et Style* un par un et de lancer la machinerie et c'est du CrystalReports standard !

34 Runtime

Il n'est pas possible à toutes les entreprises de payer une licence MS Access à leurs employés et honnêtement cela n'a guère d'intérêt... D'où le fait qu'il existe un runtime (téléchargeable gratuitement sur le site internet de Microsoft).

Déjà avant de déployer le runtime, il faut vérifier que sa base de données, fonctionne bien avec le runtime. Pour cela vous pouvez simuler le runtime en création un raccourci MS Windows avec la syntaxe suivante:

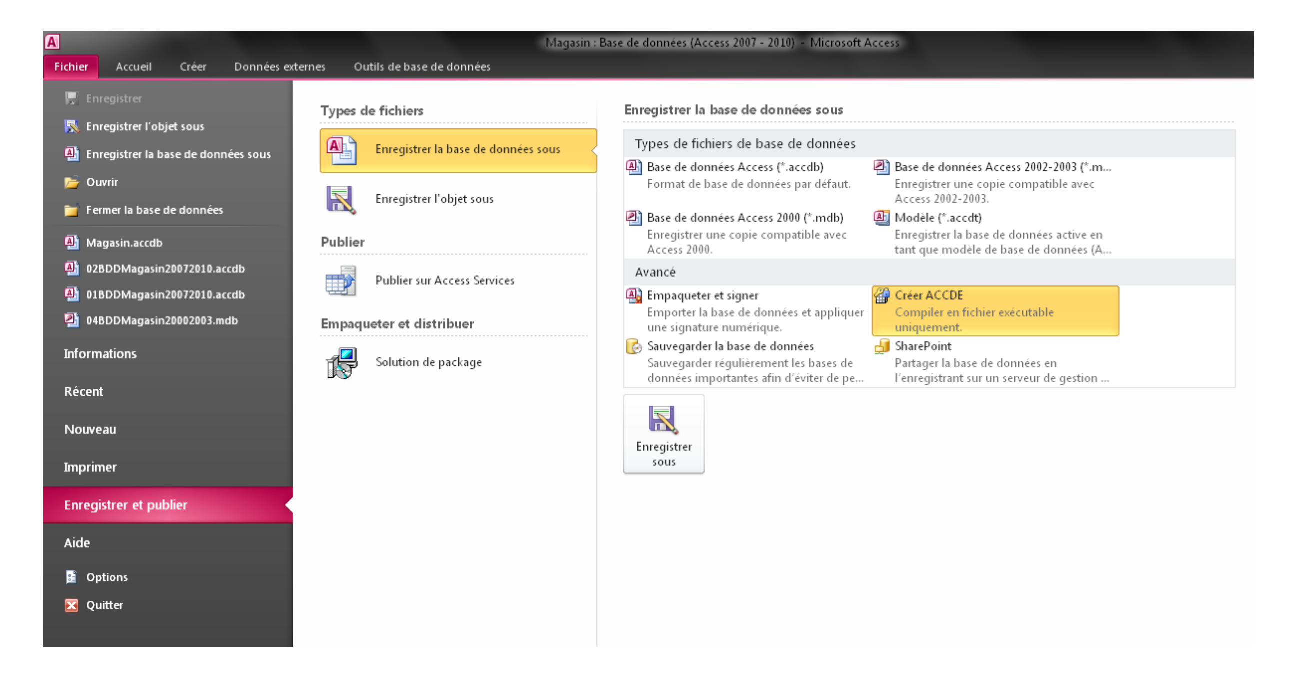
"C:\Program Files\Microsoft Office\OFFICE11\MSACCESS.EXE" _
"C:\magasin.mdb" /runtime

bon évidemment il vaut mieux travailler avec une frontable en *.mde et une dorsale *.mdb mais après c'est une histoire.

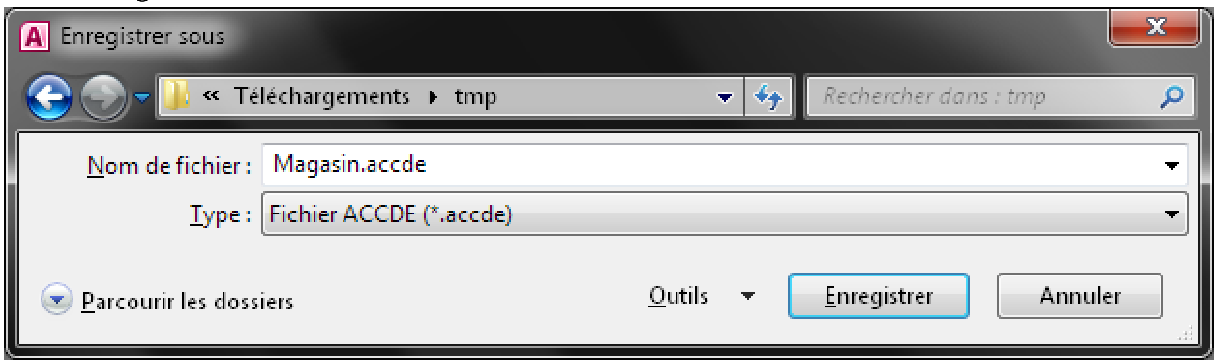
Si jamais depuis voici comment convertir une base *.accdb en fichier runtime exécutable avec MS Access 2010 (**attention**!! normalement on fait cette manipulation que lorsque l'on a séparé la base en une front-end et une back-end!!!):

1. Vous n'oubliez pas de mettre un formulaire de démarrage pour votre base de données comme nous l'avons vu plus haut

2. Vous allez dans *Fichier*/Engregistrer et choisissez l'option *Créer ACCDE*:

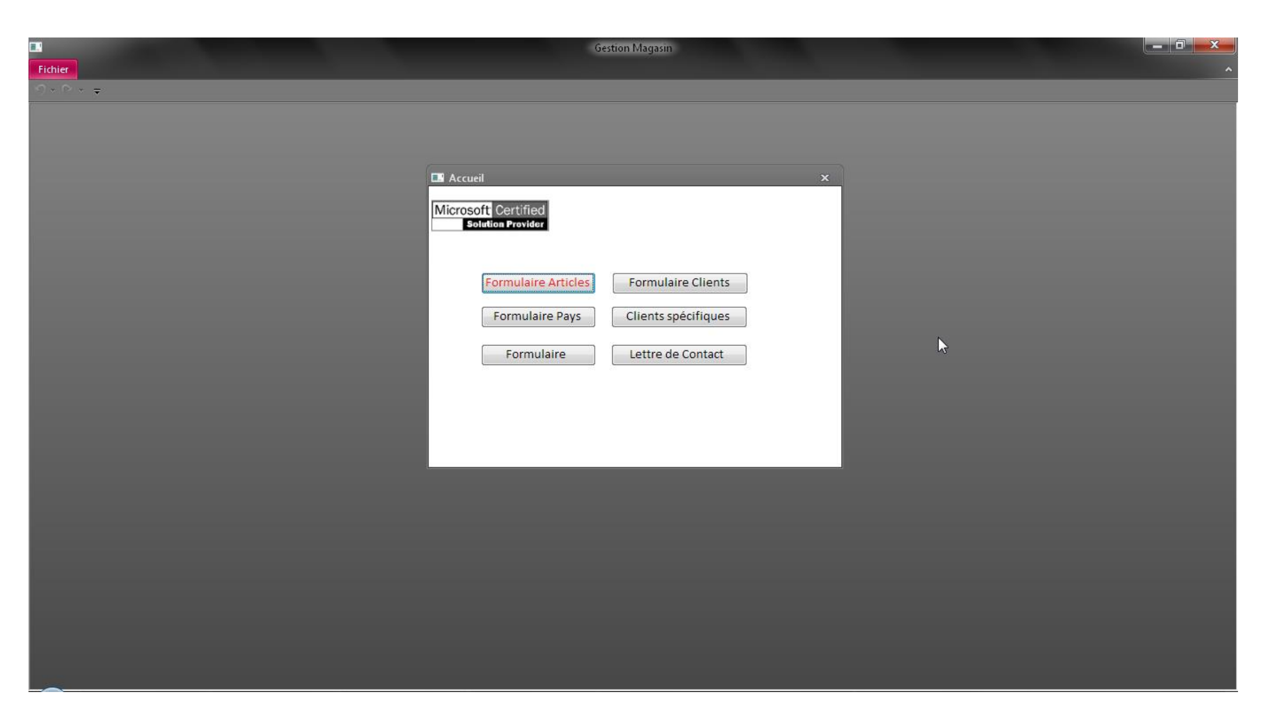


3. Vous enregistrez votre base de données à l'endroit désiré:



4. Dans le dossier avec votre fichier Magasin.accde vous renommez l'extension en *.accdr comme ci après: Magasin.accdr

   Ce qui vous donner à l'ouvertur un fichier protégé qui s'ouvre dans le runtime:



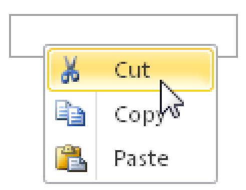
Je ne traiterai pas comment déployer et créer un fichier d'installation (*.exe) de la base de données MS Access car de toute façon en ce début de 21ème siècle, les application client n'ont aucun avenir face à la flexibilité, le design, la compatibilité multi-plateforme et la performance des bases de données internet (qui ne nécessitent qu'un navigateur et un petit serveur pour un prix négligeable).

N'oubliez que dans le mode Runtime vous n'aurez pas les filtres par formulaire!!!

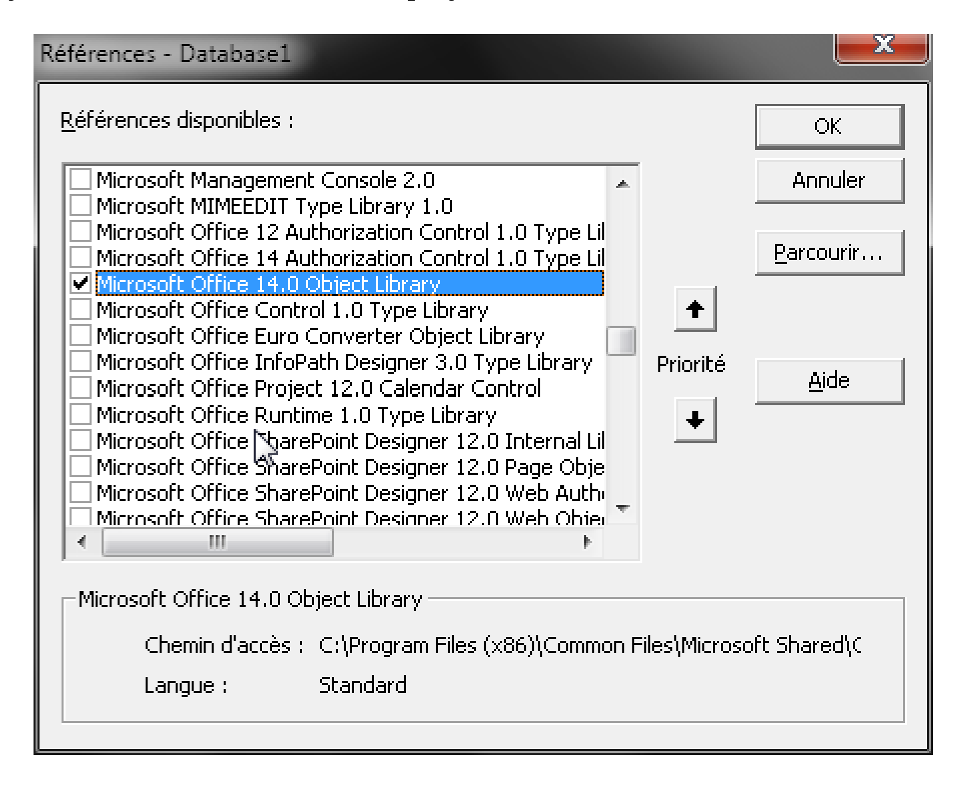
34.1 Activer le clic droit dans le runtime

Le clic droit n'est pas défaut plus disponible en mode Runtime depuis Access 2007. Donc outre la possibilité classique d'associer des boutons à la barre d'accès rapide de la base de donnée courante et qui apparaîtront alors sur la barre d'accès rapide du runtime de chacun de vos utilisateurs, vous pouvez recréer un menu contextuel avec du code VBA.

Par exemple, pour obtenir dans le runtime un menu contextuel au clic droit avec les options Couper/Copier/Coller comme visible ici:



D'abord, ajoutez la référence suivante au projet VBA:



Ensuite, créez un module avec le code suivant:

```
Option Compare Database
Option Explicit

Private Sub CreateContextMenu()

Const strMenuName As String = "Form1_CommandBar"

Dim cbar As CommandBar
Dim bt As CommandBarButton

    Set cbar = CommandBars.Add(strMenuName, msoBarPopup, , False)
    Set bt = cbar.Controls.Add
        bt.Caption = "Cut"
        bt.OnAction = "=fCut()"
        bt.FaceId = 21
    Set bt = cbar.Controls.Add
        bt.Caption = "Copy"
```

```
        bt.OnAction = "=fCopy()"
        bt.FaceId = 19
    Set bt = cbar.Controls.Add
        bt.Caption = "Paste"
        bt.OnAction = "=fPaste()"
        bt.FaceId = 22
End Sub

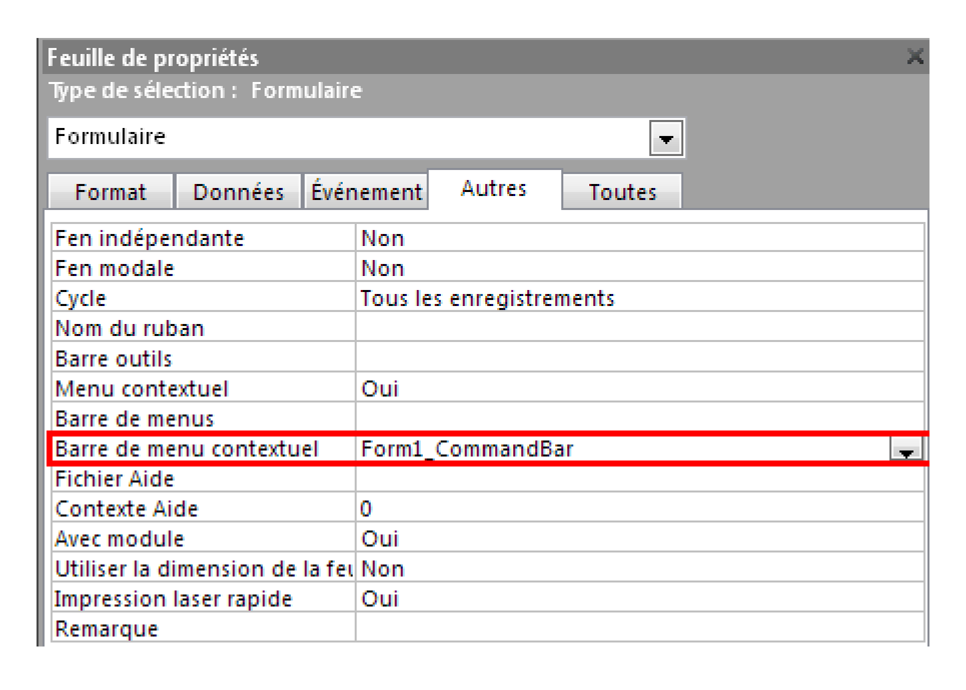
Function fCut()
On Error Resume Next
    Application.CommandBars.ExecuteMso ("Cut")
End Function

Function fCopy()
On Error Resume Next
    Application.CommandBars.ExecuteMso ("Copy")
End Function

Function fPaste()
On Error Resume Next
    Application.CommandBars.ExecuteMso ("Paste")
End Function
```

Exécutez <u>une unique fois</u> le routine principale *CreateContextMenu* afin de pouvoir par la suite l'associer à n'importe quel formulaire (si nous ne l'exécutez pas, vous ne pourrez associer ce menu à un formulaire!!).

Une fois ceci fait, dans les propriétés des formulaires de votre choix, vous changez la propriété *Barre de menu contextuel* afin d'y prendre la barre que nous venons de créer dans le module VBA:



et voilà...

35 Optimisation de bases de données

J'ai écrit ce petit chapitre par plaisir personnel et suite à une étude de cas avec un client qui devait gérer plusieurs millions de nouveaux enregistrements par jour.

Je ne suis certes pas un spécialiste dans ce domaine mais j'espère avoir quand même réussi à collecter des informations pertinentes, parfois évidentes, dans l'espoir qu'elles ne sont pas erronnées (n'hésitez pas à m'avertir le cas échéant) et d'en faire un résumé au plus propre.

Avant toute chose, je ne souhaite pas traiter de la partie électronique ou optoélectronique d'un serveur de données car il me semble quasi trivial pour analyser d'immenses masses de données, il faut privilégier des systèmes avec des mémoires vives, caches, accès lecture/écriture du HD, disques SSD, bande passante, architecture (64 bits, 32 bits…) les plus rapides et les plus fiables. On trouve déjà une littérature très abondante et de tout niveau sur le sujet.

J'ai aussi souhaité me concentrer sur un aspect qui m'intéressait un plus personnellement: les algorithmes et la complexité algorithmique de recherche de données dans les moteurs de base de données (il paraît que les disques dur utiliseraient des algorithmes équivalents).

Avant de commencer à optimiser sa base de donées et son environnement associé il me semble qu'il faut connaître les points suivants au minimum pour savoir ce qu'on veut optimiser:

- **Le modèle conceptel de données de la base** avec description des tables, des clés (sur champs ou auto-incrémentées) des contraintes d'intégrité (l'erreur type par exemple sur les forums internet est de mettre les messages dans la même table que les titres du sujet principal).

- **Le modèle physique de données de la base** avec index et chemins d'accès, tailles des blocs sur la mémoire morte (clusters).

- **Des statistiques d'utilisation** comme la taille des tables, des index, distribution des valeurs, les taux de mises-à-jour, le workload…

- **Les particularités du système hardaware** comme le parallélisme, les processeurs spécialisés, l'architecture, la connectique, la mémoire…

- **et des algorithmes** qui peuvent différer selon les systèmes…



Remarque: Parmi les techniques présentées ci-dessous, il n'y a pas de technique meilleure qu'une autre dans l'absolu.

35.1 Système FAT (File Allocation System)

Considérons une table de données. Nous y associons alors une table des matières – Table Of Content en anglais (TOC) – qui contient l'adresse du premier enregistrement et chaque enregistrement contient l'adresse de l'enregistrement suivant selon un système de chaînage. On parle parfois "d'accès direct/indirect" pour cette stratégie.

Trouver le *n*-ième enregistrement avec ce système implique alors une recherche séquentielle (complexité linéaire). C'est donc peu efficace pour rechercher une donnée et en termes de mémoire (vu que chaque enregistrement à une adresse) mais c'est la méthode de codage la plus intuitive et elle reste beaucoup utilisée dans des systèmes qui nécessitent peu de performance en termes de requêtes ou dans des systèmes développés par des amateurs éclairés.

C'est typiquement la meilleure stratégie lorsqu'on rappatrie tout le temps l'ensemble des données d'une table et ce en termes de rapidité et de mémoire… C'est également très rapide lorsqu'on insère des nouvelles données.

35.2 Système ISAM (Indexed Sequential Access Method)

Dans ce système appelé en français "organisation séquentielle indexée" (OSI), un "descripteur" (*i*-node) contient les adresses d'un certain nombre d'enregistrements équidistribués, plus l'adresse d'un bloc d'index primaire qui contient les adresses des blocs suivants. Si le descripteur fait plus qu'un certain nombre de blocs prédéfinis, on alloue un bloc pour l'adresse d'un index secondaire qui contient les adresses des prochains blocs d'index etc. Le tout a donc une structure d'arbre!

Si nous cherchons à rappatrier tous les enregistrements, nous sommes donc moins efficaces qu'une méthode FAT. Mais si nous devons aller chercher un ou plusieurs enregistrements spécifiques, nous sommes trivialement gagnants en termes de rapidité mais les *i*-node nécessitent par contre de la mémoire… A signaler aussi que l'insertion est très coûteuse lorsqu'on arrive à la fin d'un bloc: on peut alors laisser de la place libre dans chaque bloc ou réserver une zone non triée pour les insertions.

Remarque: Ce système aurait été utilisé (je n'ai pas pu vérifier) dans les débuts de Linux, Unix, le système FAT de MS Windows et les vieux SGBD! Par défaut MySQL utilise utilise la technologie ISAM jusqu'à la version 5.5.

Le fait d'avoir des blocs qui contiennent que la première valeur d'un bloc est ce que nous appelons un "index non-dense" à l'opposé des index FAT qui sont des "index denses" car chaque valeur contient une adresse.

Ainsi, une clé primaire est dans la pratique un index unique, normalement toujours trié (selon une séquence naturelle) de type non-dense ou multi-niveaux (arbre-B ou arbre-B+ que nous verrons plus loin).

35.3 Arbres

Le principe des arbres (types d'index particuliers donc!) est une version améliorée du ISAM à ma connaissance… Le principe est basé sur la stratégie du diviser pour mieux régner…

Il existe plusieurs types d'arbres dont voici les plus connus:

- Arbres B (MS Access, MySQL, NTFS)
- Arbres B+ (SQL Server, Oracle)
- Arbres-μ (électronique/mémoire flash)
- Arbres quadrants (théorie des ensembles)
- Arbres R (géomatique)
- Arbres k-d (généralisation des arbres R: 2D)

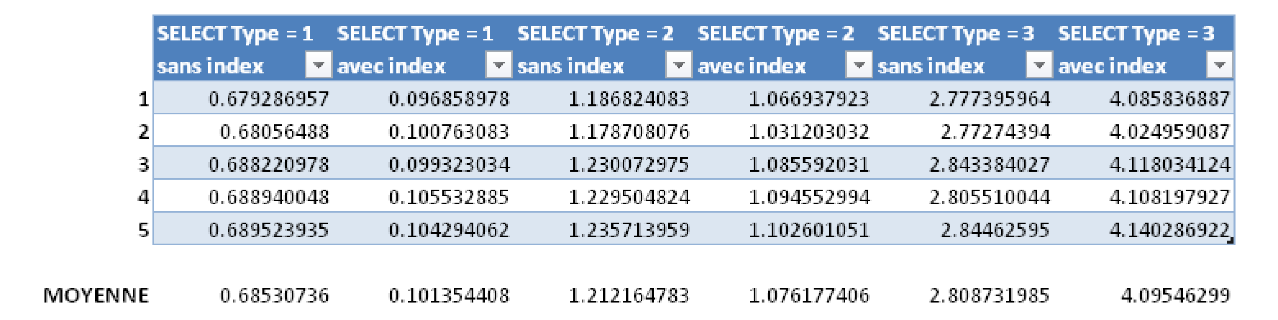
Enfin indiquons un exemple de test de performance de l'index par défaut dans MySQL 5.0 fait par un collègue avant d'aller plus loin. Le test est effectué sur une table contenant 60'000 lignes et cinq colonnes dont une nommée *Type*, contenant les valeurs 1, 2 ou 3. La valeur 1 apparaît environ 2% des fois dans la colonne, tandis que les valeurs 2 et 3 apparaissent respectivement 20% et 78% des fois (approximativement). Le test consiste à répéter 20 fois les requêtes de sélection:

SELECT * FROM NomTable WHERE Type = 1

SELECT * FROM NomTable WHERE Type = 2

SELECT * FROM NomTable WHERE Type =

et à mesurer le temps d'exécution de chacune d'entre elles. Nous avons testé ces trois requêtes avec et sans index sur la colonne *Type*, ce qui nous fait un jeu de six échantillons. Les résultats sont donnés en secondes et sont résumés dans le tableau ci-dessous:



	SELECT Type = 1 sans index	SELECT Type = 1 avec index	SELECT Type = 2 sans index	SELECT Type = 2 avec index	SELECT Type = 3 sans index	SELECT Type = 3 avec index
1	0.679286957	0.096858978	1.186824083	1.066937923	2.777395964	4.085836887
2	0.68056488	0.100763083	1.178708076	1.031203032	2.77274394	4.024959087
3	0.688220978	0.099323034	1.230072975	1.085592031	2.843384027	4.118034124
4	0.688940048	0.105532885	1.229504824	1.094552994	2.805510044	4.108197927
5	0.689523935	0.104294062	1.235713959	1.102601051	2.84462595	4.140286922
MOYENNE	0.68530736	0.101354408	1.212164783	1.076177406	2.808731985	4.09546299

Nous pouvons constater que:

1. Les deux premières colonnes nous montrent que pour des valeurs qui apparaissent très rarement dans la table (2%), une requête de sélection de ces valeurs prend nettement moins de temps avec un index sur la colonne.

2. Les troisièmes et quatrièmes colonnes nous montrent qu'aux alentour d'un taux d'apparition de 20-25%, il n'y a pas beaucoup de différence entre avoir un index ou pas.

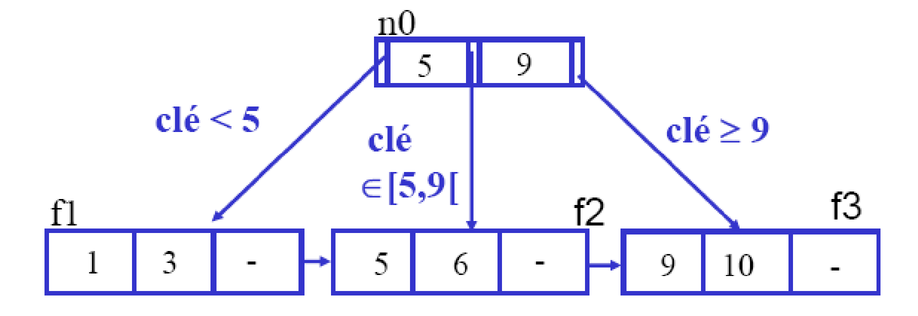
3. Enfin les deux dernières colonnes nous montrent que la situation est renversée lorsque le taux d'apparition est élevé (80%) et que dans ce cas, il vaut mieux ne pas avoir d'index.La suite viendra…

Ce montre donc qu'il faut toujours tester les performances des index avant de les considérer comme définitivement activés.

35.3.1 Arbre B

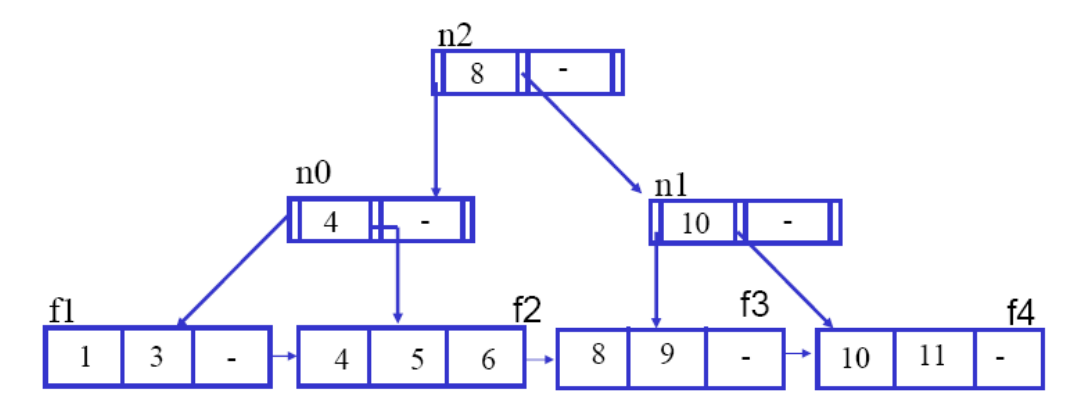
L'arbre B est simplement un organigramme dont la taille des branches n'est pas forcément égale. Nous parlons alors en toute généralité "d'arbre non balancé". C'est le type d'index qu'utilisent à ma connaissance MS Office Access, MySQL et le système NTFS. Le problème avec MS Offce Access c'est qu'au-delà de quelques centaines de millier d'enregistrements, il faut parfois aller changer des valeurs dans la base de registre pour créer des index.

Voici typiquement un exemple d'arbre (celui-ci est balancé!) à 2 niveau et à 3 feuilles (non-denses) contenant des identifiants (alvéoles) associés implicitement à des enregistrements (l'association clé-éléments est ce que nous appelons:



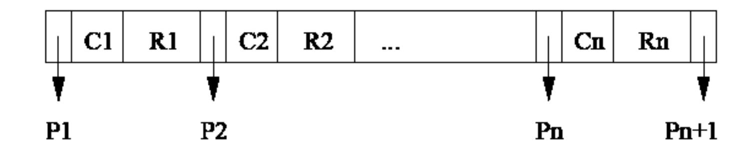
Si nous y recherchons la clé 6: $6 \in [5,9[$ donc nous allons dans f2 et $6 \in f2$.

Pour un arbre à trois niveaux:



Si nous y recherchons de la clé 9: 9≥8 donc nous allons dans n1 et comme 9 < 10 nous allons dans f3 et 9∈f3.

Chaque sous-arbre débute donc avec ce que nous appelons un "nœud". Un noeud est donc un index local, les enregistrements servant de clé (avec une table de hachage ou non), intercalés avec des pointeurs mémoire:



Le sous-arbre pointé par P2 contient ainsi tous les enregistrements dont la clé est comprise entre C1 et C2.

Considérons par exemple une table contenant 1'000'000 d'enregistrements où l'administrateur a pu définir un taille de 28 octets par entrée (taille d'une alvéole) et qu'un cluster a une taille de 4'096 octets.

Nous avons alors:

$$\frac{4096}{28}=146\ \left[alvéoles/cluster\right]=146\ \left[entrées/cluster\right]$$

Soit un total de:

$$\frac{1000000}{146}=6850\ \left[clusters\right]$$

tout en bas de l'arbre. Le niveau d'au-dessus devrait pouvoir indexer ces 6850 clusters aussi avec des alvéoles, il lui faudra donc:

$$\frac{6850}{146}=47\ \left[clusters\right]$$

Et le niveau d'au-dessus sera simplement un unique cluster avec 47 alvéoles. Il s'agit donc d'un arbre à trois niveaux. L'arbre aura donc une taille mémoire utilisée pour un millions d'enregistrement à 28 octets l'enregistrement de:

$$(47+47\cdot 146+6850\cdot 146)\cdot 28\cong 28\ [Mo]$$

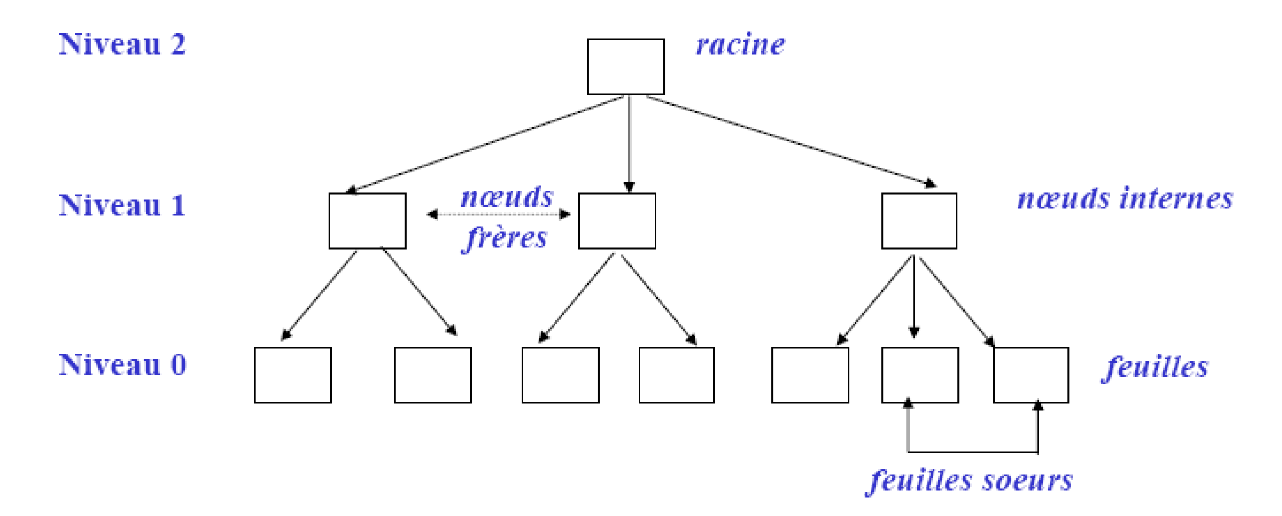
et si l'arbre était plein nous aurions:

$$146\cdot 146\cdot 146\cdot 28\cong 80\ [Mo]$$

qui correspond donc à la mémoire à allouer sur le disque pour l'index s'il venait à être plein et correspond donc à environ 3'000'000 d'enregistrements avec 3 niveaux (sur 1 niveau 146 enregistrements, sur 2 niveaux environ 20'000, sur 3 niveaux environ 3 millions et sur 4 niveax environ 450 millions).

35.3.2 Arbre B+

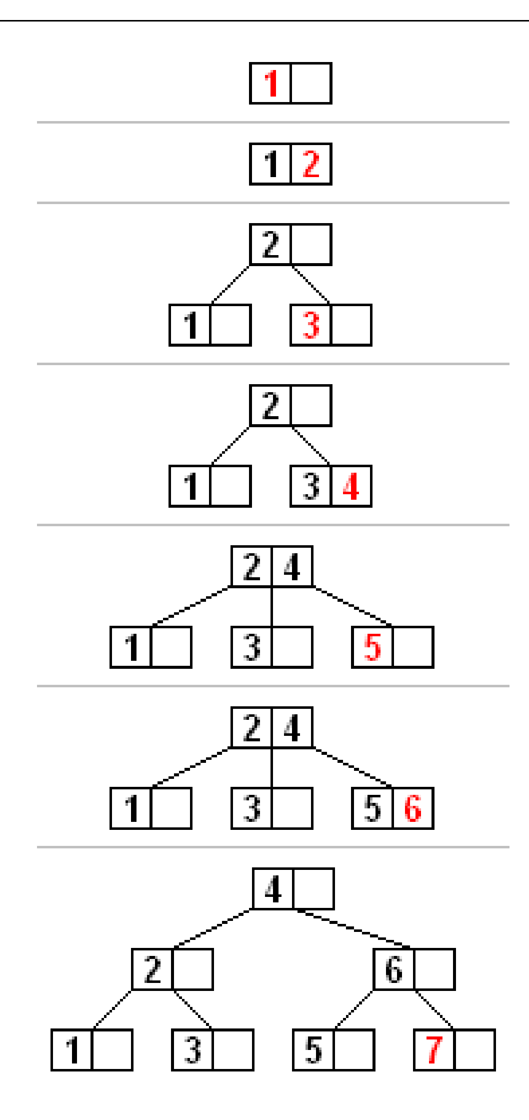
L'Arbre B+ (R. Bayer et C. McCreight, 1972), appelé également: B-tree ou arbre de Bayer. Le B+ est là pour indiqué que l'arbre est Balancé (tous les chemins partant de la racine vers une feuille ont la même longueur).



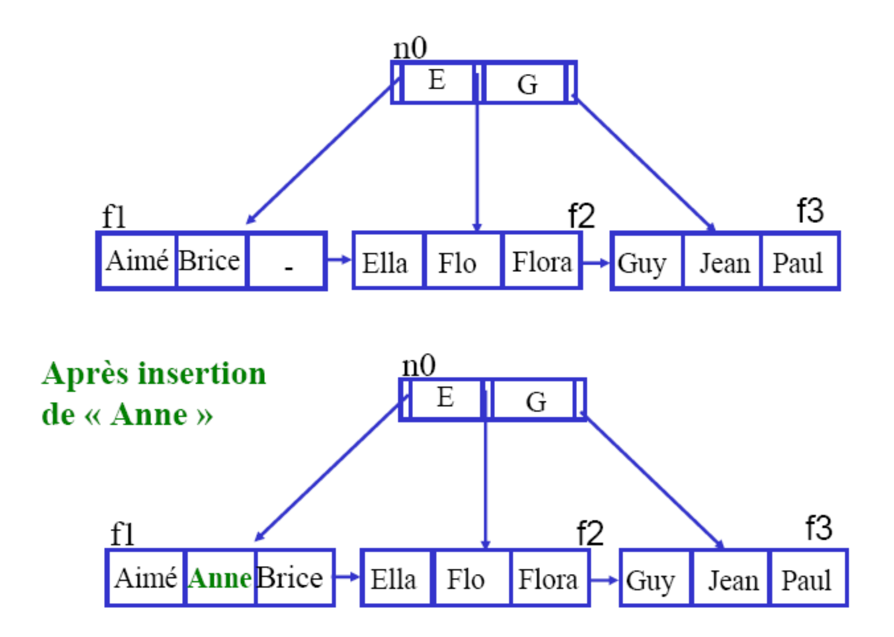
Les propriétés de l'arbre B+ sont les suivantes:

- Toutes les pages (feuilles) sont remplies au minimum à 50%
- Chaque noeud est un index local (bloc mémoire)
- Tous les noeuds dans la réalité ont la même taille (contrairement aux exemples pédagogiques qui vont suivre)
- Tous les noeuds feuilles sont au même niveau
- La hiérarchie de l'arbre grossit par la racine
- Tous les chemins de la racine aux noeuds feuilles ont la même longueur
- Contrairement à l'arbre B, la largeur et la hauteur de l'arbre est dynamique
- Contrairement à ISAM nécessite moins de pages (feuilles) de débordement (donc moins de mémoire)
- Typiquement avec 67% de remplissage en moyenne
- Nombre d'enfants d'un nœud est appelé « fan-out » et est noté m

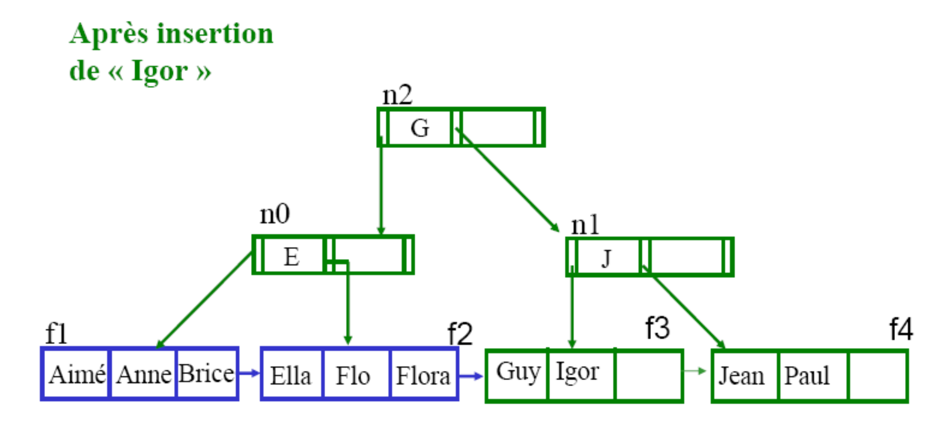
Voici comment se remplit un arbre B+ (nous y observons bien que les feuilles sont remplies au minimum à 50%):



ou avec un exemple comportant autre chose que des chiffres:



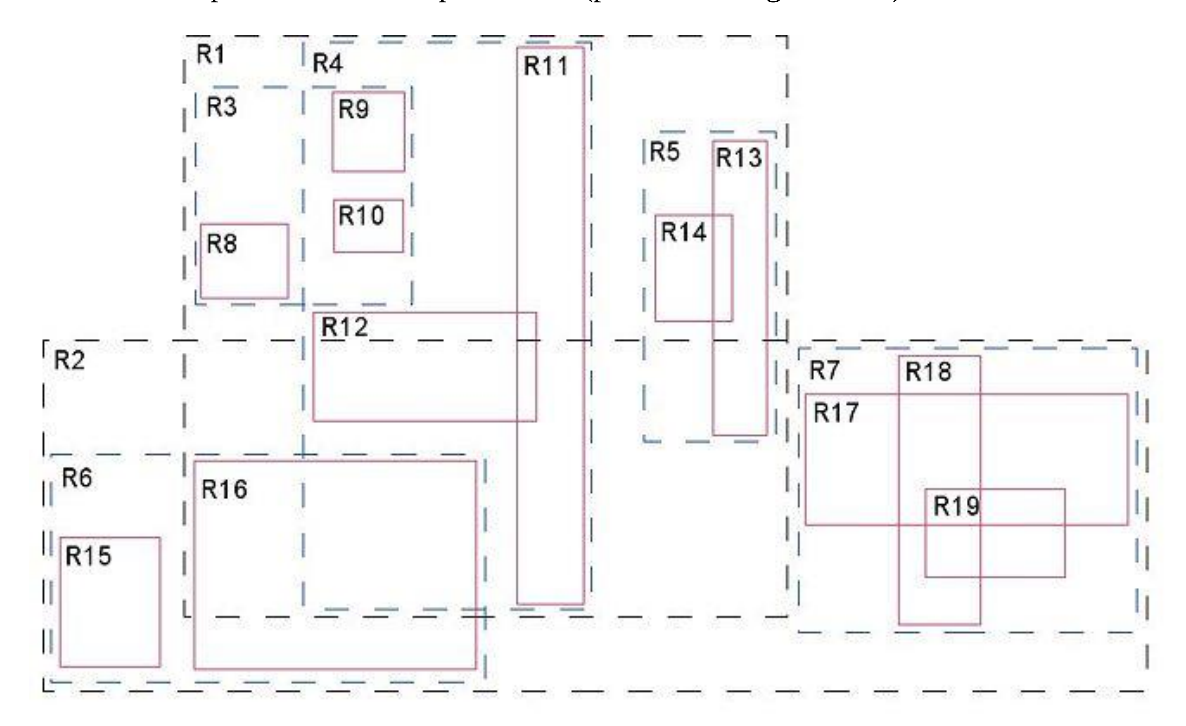
Ensuite nous rajoutons encore une personne et puisque toutes les feuilles sont pleines, il faudra faire une insertion balancée (recompalitation de la totalité de l'index):

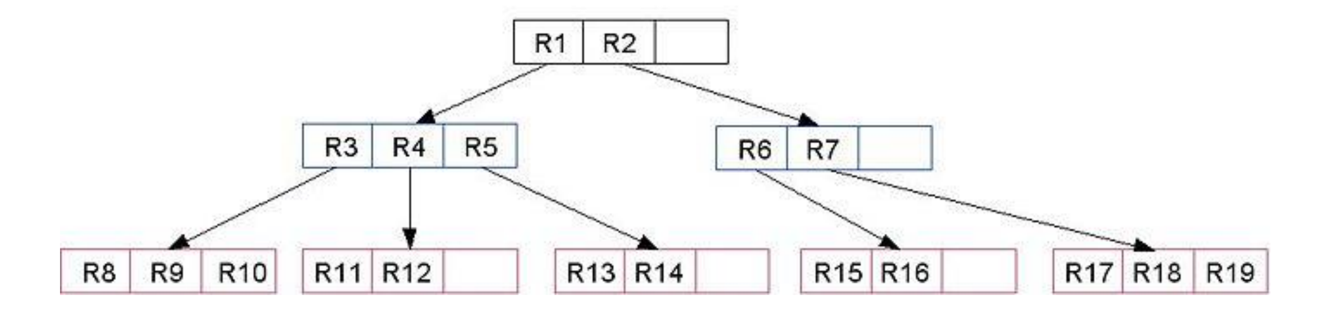


La contrainte d'occupation minimale est ainsi garantie!

35.3.3 Arbre-R

L'arbre *R* permet donc de découper des zones géographiques en ensembles et sous-ensembles et de regroupes celles-ci sous formes d'arbres afin d'accéder ensuite plus rapidement aux données. Elles sont utilisées non pas pour faire des requêtes de noms de pays, villes ou autres mais utilisées pour créer des map visuelles (pensez à Google Earth!):





36 Business Intelligence

Comme nous l'avons vu, les tableaux croisés dynamiques de Microsoft Access sont très pauvres en possibilités et ceux de Microsoft Excel ne sont guère meilleurs puisque le niveau analytique ne dépasse pas celui du collège/gymnase (Suisse) ou lycée (France). De plus il est à parier que les tableaux croisés dynamiques et graphique croisés dynamiques vont probablement disparaître de Microsoft Access d'ici quelques dizaine d'années (si ce n'est pas carrément le logiciel dans son entier!).

Si nous souhaitons rester du côté utilisateur (client) sans faire appel à des technologies serveur il va sans dire que depuis Microsoft Excel 2010 les compléments Power Pivot, Power Query et depuis 2013 les compléments Power Map et Power BI Designer vont probablement devenir en Europe l'outil de référence de B.I. au sein des entreprises (puisque c'est déjà le cas aux U.S.A.) pour des applications non académiques.

L'idée de ce chapitre est que le lecteur qui souhaite suivre une formation sur un outil B.I. sache qu'il est extrêmement pénible de faire de la BI avec Microsoft Access et même totalement inutile car les outils actuellement disponibles dans le logiciel nous le savons vont disparaître sous peu (si ce n'est pas déjà le cas au jour où vous lisez ces lignes…). Alors à quoi bon… Eh bien pour la culture général pardi et ne pas arriver comme un ignorant aux formations spécialisées sur le sujet!!!!

L'**informatique décisionnelle** (en anglais : DSS pour *Decision Support System* ou encore BI pour *Business Intelligence)* désigne scolairement les moyens, les outils et les méthodes qui permettent de collecter, consolider, modéliser et restituer les données, matérielles ou immatérielles, d'une entreprise en vue d'offrir une aide à la décision et de permettre aux responsables de la stratégie d'entreprise d'avoir une vue d'ensemble de l'activité.

En d'autres termes les enjeux de la BI sont trivialement de répondre aux questions suivantes:

- Que s'est-il passé?
- Comment cela s'est-il passé?
- Pourquoi cela s'est-il passé?
- Que va-t-il se passer?
- Que se passe-t-il en ce moment?
- Que devrait-t-il se passer?

De nos jours, les données applicatives métier sont stockées dans une (ou plusieurs) base(s) de données relationnelle(s) ou non relationnelles.

36.1 E.T.L.

Les données sont extraites, transformées et chargées dans un entrepôt de données généralement par un outil de type **ETL** (*Extract-Transform-Load*) ou en français ETC (*Extraction-Transformation-Chargement*). Il s'agit d'une technologie informatique intergicielle (comprendre middleware) permettant d'effectuer des synchronisations massives d'information d'une base de données vers une autre.

1. **Extraction:** À cette étape, nous avons extrait des données de différentes sources internes et externes. Plusieurs requêtes sont envoyées à la source des systèmes. Les données seront mises dans un **Staging Area (SA)**, le plus souvent avec la même structure que la source. Dans certains cas, nous voulons que les nouvelles données ou celles qui ont été modifié, dans ce cas, on ne retiendra que les modifications. Certains outils peuvent le faire automatiquement, c'est le **Change Data Capture (CDC)** – capture de données.

2. **Transformation:** Une fois que les données sont disponibles dans le staging, on peut facilement filtrer et trier les données en utilisant des attributs spécifiques et faire des calculs. Dans cette étape du processus ETL, on peut vérifier la qualité des données et nettoyer les données si nécessaire.

3. **Chargement:** Pour terminer le processus, les données sont chargées dans un entrepôt central (DW: Datawarehouse). De là, les données peuvent être combinées, agrégés et chargés dans des cubes ou datamart si cela est jugé nécessaire.

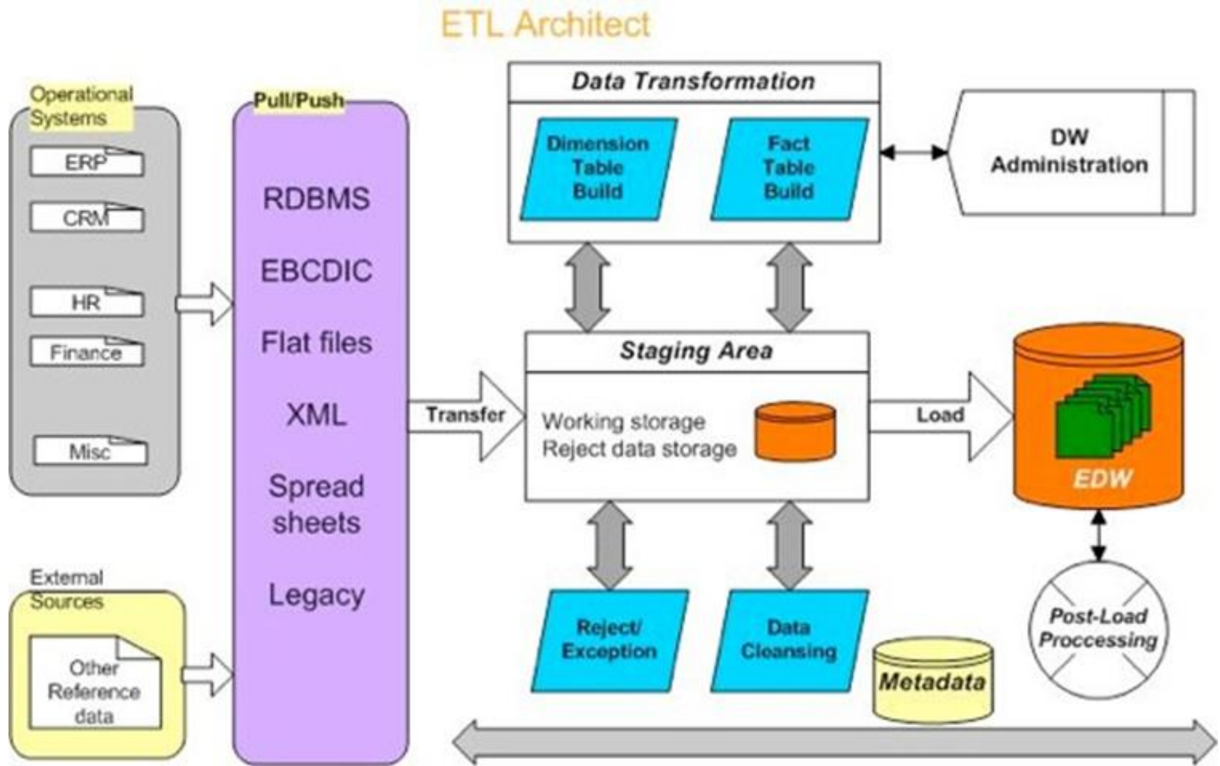


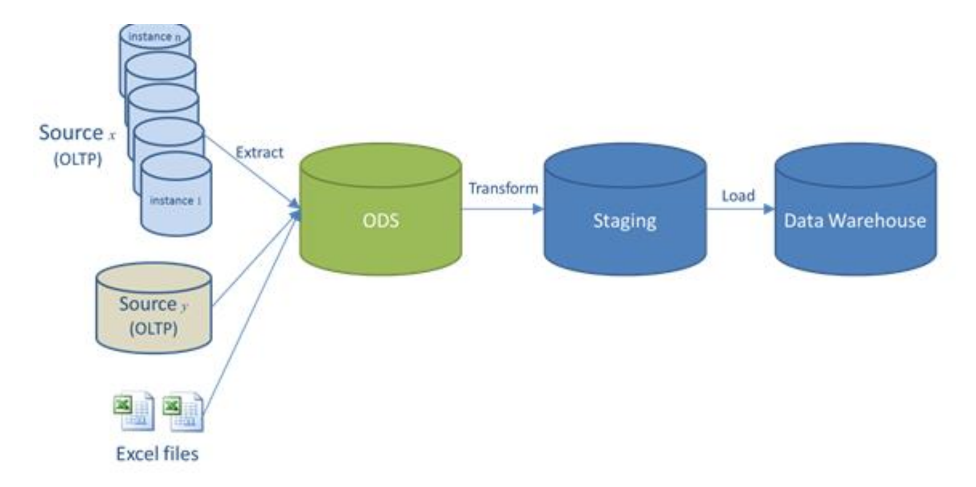
Figure 12 Architecture type d'un ETL avec Datawarehouse

36.2 Datawarehouse

Un entrepôt de données peut prendre la forme d'un **Enterprise Datawarehouse (EDW)** ou d'un **datamart**. En règle générale, le datawarehouse globalise toutes les données applicatives de l'entreprise, tandis que les datamarts (généralement alimentés depuis les données du datawarehouse) sont des sous-ensembles d'informations concernant un métier particulier de l'entreprise (marketing, risque, contrôle de gestion,…). Il va sans dire que le concept de DW respose aussi sur des bases de données relationnelles avec tout ce que cela implique.

Un **ODS** (ou Operational data store) est une base de données conçue pour centraliser les données issues de sources hétérogènes afin de faciliter les opérations d'analyse et de reporting.

L'intégration de ces données implique souvent une purge des informations redondantes. Un ODS est généralement destiné à contenir des données de niveau fin comme un prix ou le montant d'une vente, en opposition aux données agrégées tel que le montant total des ventes. Les données agrégées sont stockées dans un entrepôt de données (data warehouse).



L'Entrepôt de données intègre au sein d'une même base les informations provenant de multiples applications opérationnelles. On passe ainsi d'une vision verticale de l'entreprise, dictée par des contraintes techniques, à une vision transversale, dictée par le besoin métier, qui permet de croiser fonctionnellement les informations. L'intérêt de cette organisation est de disposer de l'ensemble des informations utiles sur un sujet le plus souvent transversal aux structures fonctionnelles (services) de l'entreprise. On dit que l'Entrepôt de données est orienté « métier », en réponse aux différents métiers de l'entreprise dont il prépare l'analyse.

Les datamart et/ou les datawarehouses peuvent permettre via l'OLAP (Online Analytical Processing) l'analyse rapide de l'activité de l'entreprise, grâce à des statistiques élémentaires recoupant des informations relatives à des activités apparemment très différentes ou très éloignées les unes des autres via des agrégats, mais dont l'étude fait souvent apparaître des dysfonctionnements, des corrélations ou des possibilités d'améliorations sensibles.

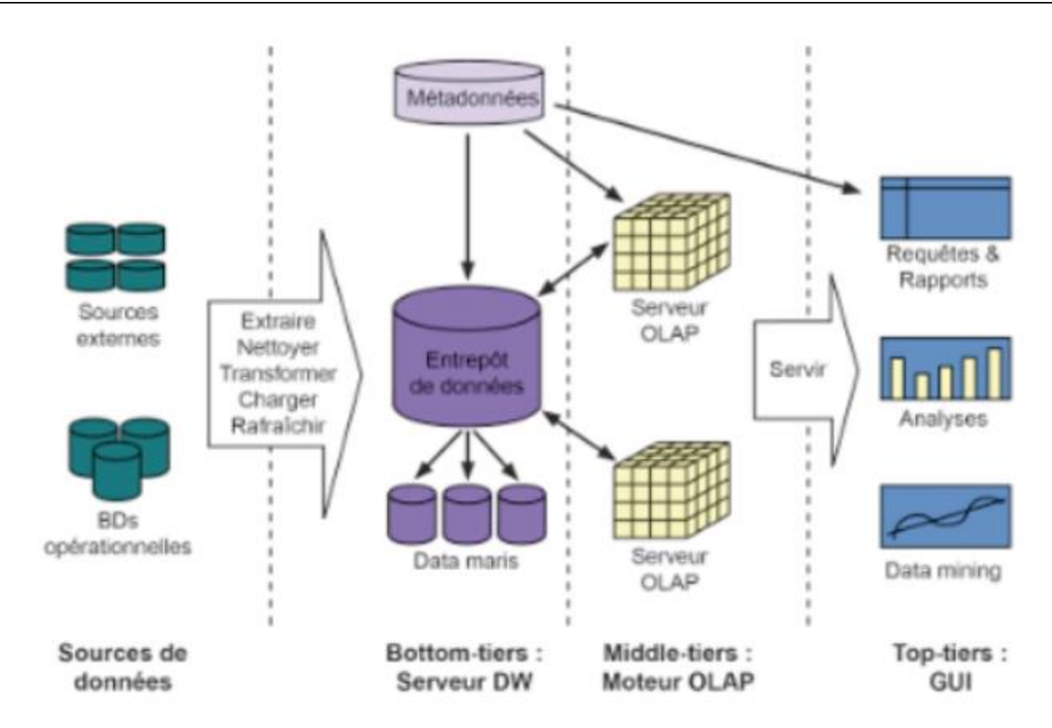


Figure 13 Ensemble des composants dans un système décisionnel (Source: 'Informatique Décisionnelle' NFE115)

L'intégration d'un datawarehouse nécessite en théorie:

- une excellente gourenance d'entreprise
- une forte activité de normalisation et de rationalisation, orientée vers la qualité
- une bonne gestion des référentiels, incluant une vérification constante de leur intégrité
- une parfaite maîtrise de la taxonomie, de la sémantique et des règles de gestion

L'**historisation** d'un Datawarehouse repose sur le principe de conservation des données (ou de nonvolatilité des données). Afin de conserver la traçabilité des informations et des décisions prises, les données une fois entrées dans l'Entrepôt sont stables, en lecture seule, non modifiables par les utilisateurs. Une même requête lancée plusieurs fois à différents moments doit ainsi restituer les mêmes résultats. Dès qu'une donnée est qualifiée pour être introduite dans l'Entrepôt de données, elle ne peut donc plus être altérée, modifiée ou supprimée (en delà d'un certain délai de purge). Elle devient, de fait, partie intégrante de l'historique de l'entreprise.

L'Entrepôt de données a une structure de données qui peut en général être représentée par un modèle de données normalisé 3FN (3 ème forme normale) pour les données de détail et/ou en étoile ou en flocon pour les données agrégées et ce dans un SGBD relationnel (notamment lorsqu'il s'agit de données élémentaires ou unitaires non agrégées). La traduction technique de ce modèle se fait souvent au sein d'un cube OLAP.

36.3 OLAP

Comme vous le savez suite suite à votre usage de Microsoft Excel, **OLAP** (On-Line Analytical Processing), désigne l'ensemble des technologies permettant la prise de décision stratégique rapide et fiable sur des données modélisées en multi dimensionnel. Ces technologies, pour mériter le label OLAP doivent posséder par consenus les propriétés suivantes:

- Conception et vue multidimensionnelle: un outil OLAP doit se baser sur un modèle multidimensionnel pour faire de l'analyse

- Transparence: la technologie utilisée, la conception ainsi que toutes les spécifications techniques doivent être invisibles à l'utilisateur final

- Accessibilité : les outils OLAP doivent permettre d'accéder les données de façon à produire de la connaissance rapide. Une information pertinente et on-time doit être fournie en tout temps

- Rapidité: les montées de charges ne doivent pas freiner l'analyste. L'outil doit pouvoir supporter de grosses requêtes (c'est la caractéristique la plus difficile à satisfaire)

- Architecture Client Serveur: pour un accès uniforme et des traitements plus rapides et plus sophistiqués!

- Dimensions génériques : essayer, autant que possible, d'avoir une unicité dans la définition des dimensions. Ne pas avoir deux dimensions client

- Gestion des matrices creuses: les matrices creuses sont des matrices qui contiennent beaucoup de zéros. En informatique, il existe des algorithmes qui utilisent cette spécificité pour optimiser le stockage de ce type de matrices. Les performances sont, en général, au rendez vous. Les outils OLAP doivent avoir cette capacité d'optimisation d'espace de stockage par la gestion des matrices creuses

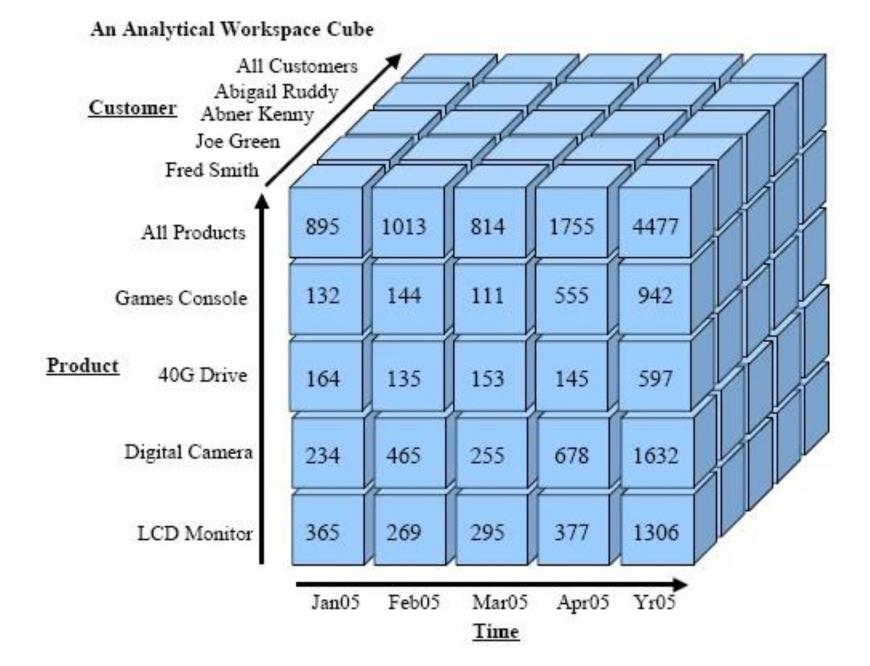
- Multi-utilisateurs: les outils OLAP sont, par définition, destinés à un accès concurrent

- Croisement inter dimensions illimité: l'utilisateur ne doit avoir aucune restriction quand au nombre de croisements qu'il fait entre les dimensions

- Intuitifs: les utilisateurs d'outils OLAP ne sont pas forcément informaticiens. Il est donc nécessaire d'offrir des solutions adaptées à leur style cognitif

- Affichage flexible: l'utilisateur doit pouvoir aisément " arranger " son résultat au format désiré et nombre quasi illimité de dimensions et de niveaux d'agrégation

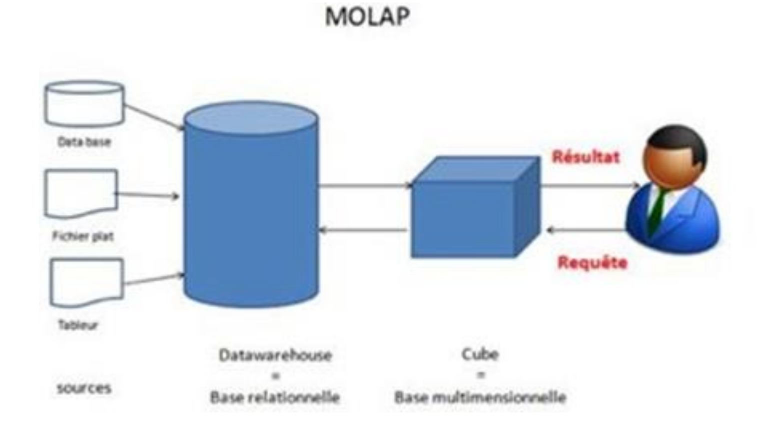
Le principe d'un cube OLAP repose sur une base de données souvent organisée en étoile. Il résultat alors dans le cas à trois critères le schéma suivant:



La technologie OLAP est souvent catégorisée en trois familles:

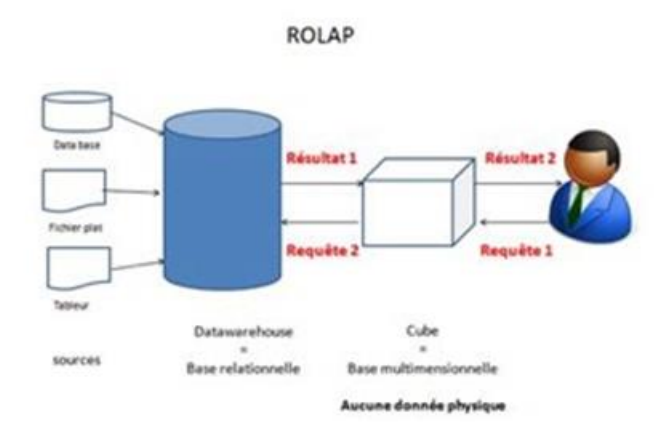
1. Le **MOLAP** pour "multi-dimensional online analytical processing" qui sont les types de cube OLAP les plus courants raison pour laquelle ils sont souvent identifiés simplement par le terme OLAP tout court…

   C'est une forme d'hypercube qui permet de représenter les données sous la forme d'un croisement de *n* dimensions optimisé (techniques de sparsing). Le cube est créé une fois et n'est pas regénéré à chaque requête de l'analyste.

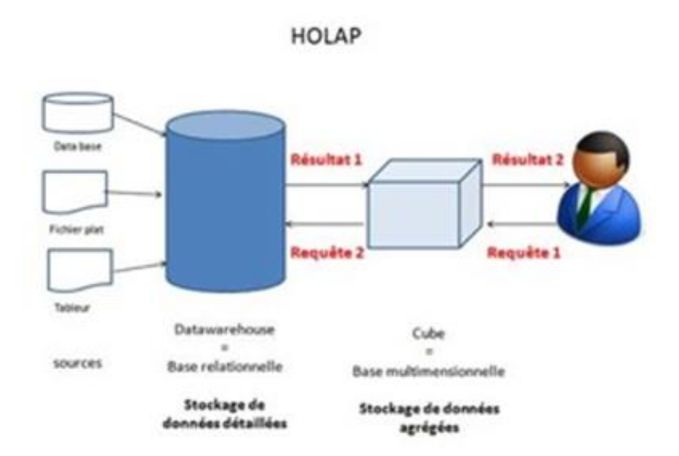
   Un exemple connu sont les vieux fichiers cubes OLAP *.cub de Microsoft Excel 2003 et antérieur qui contenaient les données mêmes du cube. Il avait pour l'avantage de ne pas nécessiter de serveur car le cube était construite qu'une seule fois.

2. Le **ROLAP** pour "relationnal online analytical processing" est simplement un MOLAP ayant des algorithmes optimisés pour reconstruire à la volée les cubes à chaque requête de l'utilisateur. Cette technologie utilisée le SQL avec du WHERE et des JOIN pour accéder directement aux données sous forme relationelle (tables liées).



Un exemple connu sont les fichiers cubes ROLAP *.cub de Microsoft Excel 2007 et antérieur liés à Microsoft SQL Server Analysis et on comprend alors pourquoi un serveur est exigé puisque le cube et reconstruit à la volée (car c'est un désavantage en termes de calculs).

3. Le **HOLAP** est simplement un hybride du MOLAP et COLAP qui s'adapte à la volée à l'usage régulier de l'utilisateur (analyse des usages et optimisations sur le long terme).



Remarque: Il y a aussi des dérivées par corps de métier comme le **WOLAP** (Web-based OLAP), le **DOLAP** (Desktop OLAP), le **RTOLAP** (Real-Time OLAP), etc.

37 Limites MS Office Access

37.1 Limites MS Access 2000+2002+2007+2010

En anglais par flemme de traduction (il n'y pas d'évolutions entre la version 2003 et 2010):

General

Attribute	Maximum
Access database (.accdb) file size	2GB, minus the space needed for system objects **Note** NOTE: Although the maximum size for a single database file is 2GB, you can work around this limitation by using a split database. A front-end database file can point to thousands of back-end database files, each of which could be as large as 2GB. For more information, see the topic, Split a database.
Number of objects in a database	32'768
Number of modules (including forms and reports that have the HasModule property set to True)	1'000
Number of characters in an object name	64
Number of characters in a password	20
Number of characters in a user name or group name	20
Number of concurrent users	255

Table

Attribute	Maximum
Number of characters in a table name	64
Number of characters in a field name	64
Number of fields in a table	255
Number of open tables	2'048; the actual number might be smaller because of tables opened internally by Access
Table size	2GB minus the space needed for the system objects 1GB in Access 2000
Number of characters in a Text field	255
Number of characters in a Memo field	65'535 when entering data through the user interface; 2 GB of character storage when entering data programmatically
Size of an OLE Object field	1 GB
Number of indexes in a table	32
Number of fields in an index	10
Number of characters in a validation message	255
Number of characters in a validation rule	2'048
Number of characters in a table or field description	255
Number of characters in a record (excluding Memo and OLE Object fields) when the UnicodeCompression property of the fields is set to Yes	4'000 2'000 in Access XP
Number of characters in a field property setting	255

Query

Attribute	Maximum
Number of enforced relationships	32 per table, minus the number of indexes that are on the table for fields or combinations of fields that are not involved in relationships*
Number of tables in a query	32*
Number of joins in a query	16*
Number of fields in a recordset	255
Recordset size	1 gigabyte
Sort limit	255 characters in one or more fields
Number of levels of nested queries	50*
Number of characters in a cell in the query design grid	1'024
Number of characters for a parameter in a parameter query	255
Number of AND operators in a WHERE or HAVING clause	99*
Number of characters in an SQL statement	Approximately 64'000*

*Maximum values might be lower if the query includes multivalued lookup fields.

Form and report

Attribute	Maximum
Number of characters in a label	2'048
Number of characters in a text box	65'535
Form or report width	22 in. (55.87 cm)
Section height	22 in. (55.87 cm)
Height of all sections plus section headers (in Design view)	200 in. (508 cm)
Number of levels of nested forms or reports	7 3 in Access 2000
Number of fields or expressions that you can sort or group on in a report	10
Number of headers and footers in a report	1 report header/footer; 1 page header/footer; 10 group headers/footers
Number of printed pages in a report	65,536
Number of controls and sections that you can add over the lifetime of the form or report	754
Number of characters in an SQL statement that serves as the Recordsource or Rowsource property of a form, report, or control (both .accdb and .adp)	32'750

Macro

Attribute	Maximum
Number of actions in a macro	999
Number of characters in a condition	255
Number of characters in a comment	255
Number of characters in an action argument	255

38 Nouveautés MS Office Access

On m'a parfois demandé de communiquer la liste des nouveautés des différentes versions sur lesquelle je donne des cours. Voici donc une énumération non exhaustive:

Pour les nouveautés de MS Access 2000 par rapport à 97:

- Presse-papier pour copier/coller multiple
- Nouvelle fonctionnalité de personnalisation des barres d'outils
- Menus évolutifs
- Envoi d'e-mail directement depuis les applications
- Historique des derniers fichiers ouverts
- Options d'ouverture dans la boîte de dialogue d'Ouverture
- Enregistrement en une version antérieure d'une base de données
- Repercussion automatique des noms des champs dans les objets dépendants
- Formatage conditionnel de champs dans les formulaires
- Export de données vers Excel par simple cliquer/glisser des tables ou requêtes
- Impression du diagramme des relations
- Option de compression à la fermeture
- Page d'accès de données web
- Interopérabilité de Microsoft SQL Server

Pour les nouveautés de MS Access XP (2002) par rapport à 2000:

- Nouvelle interface et aide en ligne
- Impression par *.mdi
- SmartTags (balises actives)
- Nouvelles options de collage spécial
- Création facilitée de certificats pour les macros
- Nouvelle option d'export au format XML

Pour les nouveautés de MS Access 2003 par rapport à XP:

- Intégration avec WSS
- Options de vérification des erreurs
- Affichage des liaisons (interdépendances) entre objets
- Sauvegarde de la base de données sous un autre nom
- Propagation des propriétés des champs
- Amélioration du tri des contrôles listbox et combobox
- Aide contextuelle dans les vues SQL

Pour les nouveautés de MS Access 2007 par rapport à 2003:

- Nouvelle Interface (onglets et rubans, rubans contextuels, galeries, super infobulles, menu Office)
- Visualisation de raccourcis claviers par la touche (Alt)
- Changement du thème de couleur de visualisation de l'interface
- Barre d'accès rapide et personnalisation pour un modèle ou le logiciel
- Personnalisation des rubans avec le langage RibbonX

- Activation de l'onglet Développeur
- Listing des documents récents fixable
- Nouveau format de fichier (*.accdb)
- Navigation par onglets (désactivation pour revenir en mode fenêtres)
- Nouveau visuel dans le MCD (schéma relationnel)
- Nouvelles options de filtrage dans les tables
- Nouveaux modèles de couleurs pour les formulaires et rapports
- Nouveau type de données et contrôle de pièces jointes
- Nouvel assistant liste de choix avec valeurs de sélection multiples (champs Objective Date Picker automatique avec les champs de type Date en mode table
- Utilisation des Layouts pour le regroupement de contrôles dans les formulaires et rapports et Gridlines
- Nouveau format de contrôle texte RTF
- Nouvelles mises en forme automatiques (autoformats)
- Rédaction directe des arguments des actions dans les objets Macros
- Affichage de la totalité des actions de macros
- Suppression du Workgroup Security Manager pour les fichiers accdb
- Suppression la Réplicaton pour les fichiers accdb
- Suppressions des formulaires Web pour les fichiers accdb
- Envoi de requête de saisie par e-mail et gestion des retours de données
- Suppression du contrôle de liste déroulante pour la recherche
- Disparition du champ de saisie de critères de filtrage au clic droit de la souris dans les champs de tables et formulaires
- Format comptabilité plus possible
- Option d'historique dans les champs mémo
- Liaison SharePoint n'est plus possible avec les vues (affichages) → importe toute la table!
- Le code VBA concernant les connexions ADODB doit être réécrit
- Nouvelles commandes VBA d'export de tables vers SharePoint
- La commande VBA acExportDelim exporte indépendamment des paramètres régionaux de l'ordinateur avec une "," comme délimiteur, il faut obligatoirement définir un modèle d'export pour avoir le ";"
- Suppression du fonctionnement correct des auto-liaisons (à priori...)

Pour les nouveautés de MS Access 2010 par rapport à 2007:

- Nouveau menu Office étendu (backstage menu)
- Personnalisation des rubans (sans RibbonX avec Import/Export de l'UI)
- Nouvelle catégorie de SmartArts (SmartArts avec images)
- Nouvelle balise active au copier/coller
- Création de macros de données dans les tables
- MS Access Services (version serveur)
- Vérificateur de compatibilité avec publication web
- Refonte complète du constructeur de macros
- Ne gère plus du tout les propriétés (description) de tables/requêtes/rapports ou macros importées ayant plus de 255 caractères
- Utilisation et création de modèles de "composants d'application"
- Utilisation et création de modèles de "champs de données"
- Suppression de l'ActiveX de contrôle de calendrier

- Nouveau contrôle de navigateur web (très utile pour l'aperçu des PDF dans les formulaires)
- Nombreuses nouvelles propriétés de formatage (typiquement pour les boutons)
- et autres pas encore découvertes…

Pour les nouveautés de MS Access 2013 par rapport à 2010:

- Suppression des tableaux et graphiques croisés dynamiques
- Suppression des options de réplication

38.1 Propositions de nouveautés ultérieures

Bon... comme pour dans mon livre pour Microsoft Project... je commence à perdre patience quant à l'absence de nouveautés conséquentes ces dix dernières années concernant ce logiciel. Donc voici une liste de nouveautés attendues:

- Proposer une version pour Macintosh (Apple)
- Revoir tout l'IDE du VBAE (numérotation des lignes, groupements de structures de codes, codage dynamique, etc.)
- Permettre d'utiliser l'ensemble du standard SQL comme le fait Oracle avec les fonctions analytiques par exemple
- Création d'un formulaire de type Calendrier avec une grande multitude de choix d'affichages
- Création d'un planning de Gantt avec WBS, navigation dans le temps, replanification par glisser-déplacer, etc.
- Permettre de contrôler plus en finesses les couleurs des onglets dans les formulaires
- Retour et modernisation des tableaux et graphiques croisés dynamiques
- retour d'une gestion avancée et native de la sécurité des utilisateurs (et dont la granularité va jusqu'aux champs eux-mêmes!!!)
- Pouvoir aligner verticalement du texte dans un champ ou label

39 Raccourcis claviers

Vous trouvez que l'utilisation du clavier est parfois plus rapide que celle de la souris ? Les touches de raccourci vous permettent d'éviter les menus et d'exécuter des commandes directement. Vous pouvez utiliser les touches de raccourci à de nombreuses fins dans Access, pour accéder aux commandes et aux boutons de la barre d'outils aussi bien que pour insérer la date du jour. Les touches de raccourci sont parfois répertoriées en regard des noms de commandes dans les menus Access. Par exemple, dans le menu Fichier, la commande Imprimer indique le raccourci CTRL+P.Pour obtenir la liste exhaustive des raccourcis, demandez de l'aide au Compagnon Office. Dans Access 2000 ou dans n'importe quelle autre application Office 2000, appuyez sur F1 pour afficher le Compagnon, puis tapez touches de raccourci dans la zone de texte. Voici quelques-unes des touches de raccourci les plus utiles de Access:

TOUCHES DE RACCOURCIS DIVERSES

F2	Pour afficher l'adresse complète du lien hypertexte sélectionné
F7	Pour vérifier l'orthographe
MAJ+F2	Pour ouvrir la zone Zoom afin de faciliter la saisie d'expressions et autre texte dans des zones de saisie réduites. Pour MS Access 2007 et les champs Mémo (Textes Longs) avec l'option RTF activée ne permet cependant pas d'utiliser correctement cette nouvelle fonctionalité. Il faut passer par un formulaire.
ALT+ENTREE	Pour afficher une feuille des propriétés en mode Création.
ALT+F4	Pour quitter Microsoft Access, fermer une boîte de dialogue ou fermer une feuille des propriétés
CTRL+F2	Pour appeler un Générateur
F11	Afficher la fenêtre de la base de données
CTRL+Z	Annuler
ÉCHAP ÉCHAP (appuyez deux fois sur ÉCHAP)	Annuler les modifications apportées à l'enregistrement en cours
ÉCHAP	Annuler les modifications apportées au champ en cours
ALT+F11	Basculer entre Visual Basic Editor et la fenêtre active précédente
CTRL+V	Coller
CTRL+C	Copier
CTRL+S	Enregistrer
CTRL+P	Imprimer
CTRL+;	Insérer la date du jour
CTRL+'	Insérer les données du même champ dans l'enregistrement précédent
CTRL+:	Insérer l'heure courante
CTRL+ENTRÉE	Insérer un retour chariot dans un champ de type mémo ou texte
CTRL+O	Ouvrir une base de données existante

CTRL+N	Ouvrir une nouvelle base de données
CTRL+F	Rechercher et remplacer
CTRL+F11	Pour basculer entre une barre de menus personnalisée et une barre de menus intégrée

LES TOUCHES DE RACCOURCI POUR LES OPERATIONS DE DONNÉES

Ctrl+-	Supprime la fiche active d'un formulaire ou la ligne active d'une table
CTRL++	Création d'un nouvel enregistrement dans un formulaire ou dans une table

LES TOUCHES DE RACCOURCI POUR LES OPERATIONS DE FENETRE

CTRL+./CTRL+,	Pour changer le mode d'affichage des requêtes (Ctrl+. exécute la requête, Ctrl+, remet la requête en mode création)
F11 ou ALT+F1	Pour amener la fenêtre Base de données au premier plan
CTRL+F6	Pour parcourir les fenêtres (onglets) ouvertes (pour passer de l'une à l'autre en d'autres termes)
ENTREE	Pour rétablir la taille de la fenêtre réduite sélectionnée lorsque toutes les fenêtres sont réduites.
CTRL+F8	Pour activer le mode Redimensionner de la fenêtre active lorsqu'elle n'est pas agrandie ; appuyez sur les touches de direction pour redimensionner la fenêtre.
ALT+ESPACE	Pour afficher le menu Système
MAJ+F10	Pour afficher le menu contextuel.
CTRL+W ou CTRL+F4	Pour fermer la fenêtre active.
ALT+F11	Pour basculer entre Visual Basic Editor et la fenêtre active précédente.
ALT+MAJ+F11	Pour basculer entre Microsoft Script Editor et la fenêtre active précédente

LES TOUCHES DE RACCOURCI POUR MODIFIER UN CONTROLE

MAJ+ENTREE	Pour ajouter un contrôle à une section.
CTRL+C	Pour copier le contrôle sélectionné vers le Presse-papiers.
CTRL+X	Pour couper le contrôle sélectionné et le placer dans le Presse-papiers.
CTRL+V	Pour coller le contenu du Presse-papiers dans le coin supérieur gauche de la section sélectionnée.
CTRL+DROITE	Pour déplacer le contrôle sélectionné vers la droite.
CTRL+GAUCHE	Pour déplacer le contrôle sélectionné vers la gauche.
CTRL+HAUT	Pour déplacer le contrôle sélectionné vers la haut.
CTRL+BAS	Pour déplacer le contrôle sélectionné vers le bas.
MAJ+BAS	Pour augmenter la hauteur du contrôle sélectionné.
MAJ+DROITE	Pour augmenter la largeur du contrôle sélectionné.

MAJ+HAUT	Pour réduire la hauteur du contrôle sélectionné.
MAJ+GAUCHE	Pour réduire la largeur du contrôle sélectionné.

LES TOUCHES DE RACCOURCI POUR L'UTILISATION DU MODE "CREATION"

F2	Pour basculer entre le mode Modification (quand le point d'insertion est affiché) et le mode Déplacement.
F5	Pour basculer du mode Création de formulaire au mode Formulaire.
F6	Pour basculer entre les parties inférieure et supérieure d'une fenêtre (mode Création des tables, macros et requêtes et la fenêtre Filtre/tri avancé uniquement).

LES TOUCHES DE RACCOURCI POUR RECHERCHE OU REMPLACER DU TEXTE OU DES DONNEES

CTRL+F	Pour ouvrir l'onglet Rechercher de la boîte de dialogue Rechercher et remplacer (modes Feuille de données et Formulaire uniquement).
CTRL+H	Pour ouvrir l'onglet Remplacer de la boîte de dialogue Rechercher et remplacer (modes Feuille de données et Formulaire uniquement).
MAJ+F4	Pour trouver l'occurrence suivante du texte spécifié dans la boîte de dialogue Rechercher et Remplacer lorsque la boîte de dialogue est fermée (modes Feuille de données et Formulaire uniquement).

LES TOUCHES RACCOURCIS POUR L'UTILISATION D'UNE ZONES DE LISTE

F4 ou ALT+BAS	Pour ouvrir une zone de liste modifiable
F9	Pour actualiser le contenu d'un champ Liste de choix d'une zone de liste ou d'une zone de liste modifiable
BAS	Pour descendre d'une ligne
PG.SUIV	Pour descendre d'une page
HAUT	Pour monter d'une ligne
PG.PREC	Pour monter d'une page
TAB	Pour quitter la zone de liste modifiable ou la zone de liste

LES TOUCHES RACCOURCIS POUR L'IMPRESSION ET LA SAUVEGARDE

CTRL+P	Pour imprimer l'objet sélectionné ou actif
CTRL+S ou MAJ+F12 ou ALT+MAJ+F2	Pour enregistrer un objet de base de données
F12 ou ALT+F2	Pour ouvrir la base de données Enregistrer sous

LES TOUCHES DE RACCOURCI POUR L'OUVERTURE DE BASES DE DONNEES

CTRL+N	Pour ouvrir une nouvelle base de données
CTRL+O	Pour ouvrir une base de données existante

LES TOUCHES DE RACCOURCI POUR L'AFFICHAGE DE L'AIDE

F1	Pour afficher l'aide de l'Assistant Office et de Microsoft Access, l'aide intuitive relative à la propriété, au contrôle, à l'action de macro ou au mot réservé Visual Basic sélectionné ; ou les messages d'alerte comportant un bouton Aide.
MAJ+F1	Pour afficher des Info-bulles ; après avoir appuyé sur MAJ+F1, placez le pointeur sur la commande de menu, le bouton de barre d'outils ou une région de l'écran, puis cliquez le bouton de la souris.
MAJ+F1	Pour afficher des info-bulles dans une boîte de dialogue, après avoir sélectionné l'option à l'aide du clavier.